Continuous Covering on Networks: Strong Mixed Integer Programming Formulations

Mercedes Pelegrín[1,a], Liding Xu[1,a,*]

[a]Laboratoire d'Informatique (LIX), École Polytechnique, 91128 Palaiseau, France

Abstract

Covering problems are well-studied in the domain of Operations Research, and, more specifically, in Location Science. When the location space is a network, the most frequent assumption is to consider the candidate facility locations, the points to be covered, or both, to be discrete sets. In this work, we study the set-covering location problem when both candidate locations and demand points are continuous sets on a network. This variant has received little attention, and the scarce existing approaches have focused on particular cases, such as tree networks and integer covering radius. Here we study the general problem and present a Mixed Integer Linear Programming formulation (MILP) for networks with edges' lengths no greater than the covering radius. The model does not lose generality, as any edge not satisfying this condition can be partitioned into subedges of appropriate lengths without changing the problem. We propose a preprocessing algorithm to reduce the size of the MILP, and devise tight big-M constants and valid inequalities to strengthen our formulations. Moreover, a second MILP is proposed, which admits edges' lengths greater than the covering radius. As opposed to existing formulations of the problem (including the first MILP proposed herein), the number of variables and constraints of this second model does not depend on the lengths of the network's edges. This second model represents a scalable approach that particularly suits real-world networks, whose edges are usually greater than the covering radius. Our computational experiments show the strengths and limitations of our exact approach on both real-world and random networks. Our formulations are also tested against an existing exact method.

Keywords: Continuous Facility Location, Location on Networks, Set-Covering Location Problem, Mixed Integer Programming

1. Introduction

Covering in Operations Research refers to the optimization problem of deciding the location of facilities to "cover" the points of the so-called demand set, which should fall within the radius coverage of at least one of the installed facilities. This classic problem finds applications in many different domains, including health care [1], surveillance of transport networks [2], computer networks security [3], crane location for construction [4], military evacuation systems [5], homeland defense [6], and urban air mobility [7].

Covering problems have taken many forms in the literature. A rough classification distinguishes between *maximal covering* location and *set-covering* location problems. The former aims at maximizing the covered demand with a fixed number of facilities (see e.g. [8]), while the latter seeks to minimize the number of

*Corresponding author

Email addresses: pelegringarcia@lix.polytechnique.fr (Mercedes Pelegrín), liding.xu@polytechnique.edu (Liding Xu)

[1]The authors have contributed equally.

Preprint submitted to Elsevier *March 14, 2022*

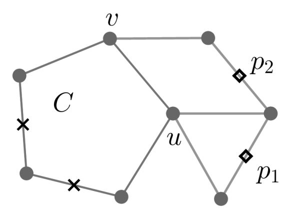


Figure 1: Two cycle coverage points with respect to a cycle C of five nodes

installed facilities to cover all the demand (see e.g. [9]). In these classic works [8, 9], the problem is defined on a network and both demand points and candidate facility locations are at nodes. Most of the variants of network covering studied afterward consider at least one of these two sets to be discrete, see the reviews [10, 11] and the references therein. However, this assumption corresponds to ideal but usually unrealistic scenarios (the reader is referred to the examples of real applications of the above paragraph). Some works addressing network covering with continuous sets of both candidate locations and demand points are [12, 13, 14], for maximal covering, and [15, 16, 17], for set-covering. We focus on the latter variant, which we call the continuous set-covering problem.

Gurevich et al. [15] presented an algorithm to compute an optimal continuous set-covering when the covering radius and the edge's lengths are natural numbers. This algorithm is polynomial time for the class of networks satisfying that every non-separable component is either an edge, a simple cycle, or a simple cycle with one chord, that is, for "almost tree" networks. More recently, Fröhlich et al. [16] also studied the same version of the continuous set-covering with natural numbers. The authors presented three different approaches to solve the problem, including a Mixed Integer Linear Programming (MILP) formulation. On the other hand, Hartmann et al. [17] focused on the computational complexity of the continuous set-covering for general covering radii. They proved that, when all edges have unit length, the continuous set-covering is polynomially solvable if the covering radius is a unit fraction, and is NP-hard otherwise.

We can now formally state our problem. Consider an undirected connected network $N = (V, E, l)$, where $l : E \to \mathbb{R}_+$ is the edges' length function. We will denote $l_e := l(e)$ the length of e. The continuum of points on all edges and nodes of N is denoted with $C(N)$. The distance function $d(\cdot, \cdot)$ defines the distance between two points, which coincides with the length of a shortest path in $C(N)$ connecting them. Given $\delta > 0$, a point $p \in C(N)$ is said to δ-cover $p' \in C(N)$ (respectively, p' δ-covers p) if $d(p, p') \leq \delta$ holds. The parameter δ is called the covering radius. The continuous δ-covering location problem on N is to find a set of facility locations in $C(N)$ of minimum cardinality that δ-covers the whole network, and is formally stated next.

Definition 1.1 (Continuous Set-Covering Problem (CSCP$_\delta$))**.** The Continuous Set-Covering Problem on a network N can be expressed as the following optimization problem:

$$\min \Big\{ |\mathcal{P}| : \mathcal{P} = \{p_i\}_{p_i \in C(N)} \text{ and } \forall p \in C(N), \exists p_i \in \mathcal{P} \text{ s.t. } d(p, p_i) \leq \delta \Big\}. \tag{1}$$

A set $\mathcal{P}$ satisfying the condition within (1) is called a $\delta-$cover of N, while $\mathcal{P}^*$ minimizing (1) is a minimum $\delta-$cover.

The set $\mathcal{P}$ in Definition 1.1 can represent the locations of ambulance bases [1], surveillance cameras [2], routing servers in a network of computers [3], cranes for construction [4], aerial military medical evacuation facilities [5], aircraft alert sites for homeland defense [6], or eVTOL safety landing sites in an urban area [7].

The CSCP$_\delta$ is known to be NP-hard, see [17]. A first observation is that typical simplifications proposed for other studied variants of set-covering are not valid for the CSCP$_\delta$. In particular, finite dominating sets (FDS) identify discrete subsets of candidate locations guaranteed to contain an optimal solution. We know at least three FDS for related variants of the CSCP$_\delta$. First, Church and Meadows [18] studied the problem with demand at nodes, and identified the following points:

$$NIP := \{p \in C(N) : \ d(p, v) = \delta \ \text{ for some } v \in V\}.$$

The authors proved that $FDS_1 := V \cup NIP$ is an FDS for the network set-covering problem when the set of demand points is V and that of candidate locations is $C(N)$. Secondly, Gurevich et al. [15] studied the continuous set-covering problem when the covering radius and the edge's lengths are natural numbers. They presented a FDS for the case of uniform unit length of the edges, which can be easily extended to the case of general edges' lengths (see [16]),

$$FDS_2 := \left\{p \in C(N) : \ d(p, v) = \frac{i}{2 \cdot l_e} \ \text{ for some } e \in E \text{ and } v \in e;\ i = 0, \ldots, 2 \cdot l_e\right\}.$$

Note that FDS_2 depends on the edge's length. Lastly, Fröhlich et al. [16] proposed a different FDS for the same version of the continuous set-covering with natural numbers. The authors defined the following set of cycle coverage points:

$$CCP := \left\{p \in C(N) : \ d(p, C) := \min_{y \in C}\{d(p, y)\} = \left(\delta - \frac{l_C}{2}\right) \bmod \delta,\ p \notin C, \ \text{ for a simple cycle } C \subseteq C(N)\right\},$$

where l_C is the total length of the cycle C. Suppose that a cycle C is covered by a set of facilities. A cycle coverage point is the furthest point where a facility that contributes to cover C can be moved without compromising the coverage of the cycle (if the rest of the facilities remain unchanged). Figure 1 illustrates this idea. In the depicted example, all edges have unit length and $\delta = 2$. The figure depicts p_1 and p_2, which are CPP with respect to the cycle C of five nodes. Note that $d(p_1, C) = d(p_2, C) = 1.5 (= (2 - 5/2) \bmod 2)$. Figure 1 also depicts two locations in C (marked with symbols 'x'), which correspond to two possible feasible locations for the remaining facility needed to cover C (note that the one at the bottom only yields a covering of the cycle if p_2 is located, while the other one together with either p_1 or p_2 can completely cover C). The authors of [16] gave the following recursive definition of an FDS for the problem with natural numbers:

$$S_1 := V \cup NIP \cup CCP;$$

$$S_{j+1} := S_j \cup \{p \in C(N) : \ d(p, y) = \delta \text{ for some } y \in bd(\mathcal{A}(S_j))\};$$

$$FDS_3 := S_{|J|},$$

where $bd(\mathcal{A}(S_j))$ is the boundary of the area covered by S_j, and $J \subseteq E$ is the subset of edges to be covered. As the author explained themselves, $FDS_3 \subseteq FDS_2$. However, the cardinality of FDS_3 may be exponential in the input size, as the number of cycles in a network is in general exponential.

The example depicted in Figure 2 illustrates that none of FDS_1 and FDS_2 are FDS for the CSCP$_\delta$. A similar observation was already presented in [13] for a related problem. The figure shows eight nodes on a path, where all edges have equal length. If $l_e = 1$ for all $e \in E$ and $\delta = 1.2$, $\mathcal{P} := \{p_1, p_2, p_3\}$ is an optimal δ-cover. It can be easily observed that there is no optimal solution in which p_2 is placed either at a node or at a distance δ from some of the eight nodes, which shows that FDS_1 is not a valid FDS. On the other

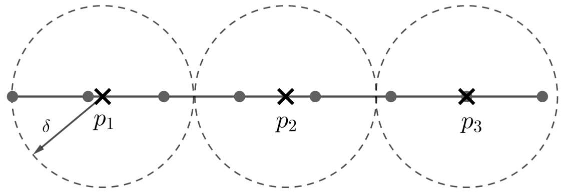


Figure 2: An instance of CSCP$_\delta$ such that not all facilities in $\mathcal{P}^*$ are at a distance δ from some node

hand, it is also easy to check that there is not a feasible solution with the three facilities located either at nodes or middle-points of edges, which proves that FDS_2 is also not a valid FDS. As opposed to FDS_2, the assumption of the edges' lengths and coverage radius being natural numbers is not fundamental in the definition of FDS_3. Conversely, FDS_3 is based on the idea of identifying those points at the "boundaries" of coverage areas, i.e., those delimiting the transition from covering/not covering a specific part of the network. Consequently, FDS_3 could be extended to the general CSCP$_\delta$. However, such an extension potentially yields sets with many more candidates, due to the recursive construction of FDS_3 based on the distance function. Note that, if δ and the edges' lengths are natural numbers, FDS_3 only contains points of the set

$$INT := \{p \in C(N) : d(p, v) \text{ is integer or half-integer for some } v \in V\}.$$

Indeed, if $\delta \in \mathbb{N}$, $NIP \subseteq INT$ is clear; $CCP \subseteq INT$ holds since the operation $(\delta - l_C/2) \bmod \delta$ only yields half integers; and $FDS_3 \subseteq INT$ then easily follows by definition. However, if $\delta \in \mathbb{R}$, the locations of the points in FDS_3 are a priori undetermined, and its cardinality increases. Take the same example depicted by Figure 1. If $\delta = 2.1$ (i.e. we increase δ just by 0.1), the CCP with respect to the cycle C of the Figure increases from two to four points.

Our main contribution is to propose an exact approach for the CSCP$_\delta$, together with tailored algorithms and strategies to tackle it. Even if this problem has been known for decades, surprisingly, only a few partial results are known for some special cases and sub-classes of networks. To the best of our knowledge, only one MILP model [16] has been proposed so far which can address the general CSCP$_\delta$. Such a model can be applied to any network whose edges do not measure more than the covering radius. This condition does not restrict the applicability of the MILP in [16], as any network can be transformed into an equivalent one that satisfies it. Here, we present an enhanced MILP formulation that relies on the same assumption as that in [16], but whose numbers of constraints and variables have smaller order of magnitude. In addition, preprocessing strategies to reduce the number of variables of the model are studied, and tailored algorithms are presented. Approaches to strengthen this formulation are also presented, including big-M constants tightening and valid inequalities. The latter includes new valid inequalities that can be seen as optimality rules for the CSCP$_\delta$, which could be used in other formulations different from those proposed in this work. The introduction of a second MILP, which is scalable concerning the edge's lengths completes the main contributions of the paper. This second MILP is an adaptation of the first one we propose, with the difference that it does not require all edges' lengths to be smaller than the covering radius. Finally, our computational experiments prove that the MILP model in [16] is not scalable. On the other hand, the preprocessing technique drastically reduces the size of the first model proposed herein. Finally, we show in the experiments that the second model we propose is superior to both the model from [16] and our first model, in terms of the solution quality and solving time.

In the proposed setting, both the candidate facility locations and the demand points are continuous sets (in particular, they coincide with $C(N)$). The problem could be defined for a subset of demand edges, $J \subseteq E$, and/or a subset of candidate locations $H \subseteq E$. The theoretical results, model, and methods described in this paper apply to such cases, after straightforward adaptation.

The rest of the paper is organized as follows. Section 2 presents useful notations and the theoretical development upon which our model is built. Then, our first MILP model is introduced in Section 3, while strategies to strengthen this model are described in the next section. The network processing algorithms that complement our MILP are detailed in Section 5. A second MILP model, which we call reduced formulation and is a modification of the first MILP, is presented in Section 6. Finally, Section 7 describes our computational experiments, and reports and analyzes the obtained results. Section 8 closes the paper with some conclusions.

2. Covering characterization

This section presents several notations, definitions, and results related to the CSCP$_\delta$. On the one hand, Proposition 2.1 gives a characterization of the $\delta-$covers of a network, which is based on the individual coverage of each edge of the network. Then, this result is refined to obtain a second necessary and sufficient covering condition in Proposition 2.2. It distinguishes between two alternative possibilities for covering each edge, namely complete or partial, and will be useful for our MILP formulation and methods. The rest of the section is oriented to characterize the so-called partial and complete covers. The idea of these sets is to delimit the areas of the network where a facility, if placed, would completely cover a given edge, and those where a facility would reach the edge (but maybe not completely covering it).

We assume that V is totally ordered by the binary relation $\preceq$. Every edge $e \in E$ has a unique representation, $e = (v_a, v_b)$, where $v_a, v_b \in V$, and $v_a \preceq v_b$. From now on, we take $e = (v_a, v_b)$ indifferently as a continuum in $C(N)$ or as an edge ending at v_a, v_b. We extend the edges' length function to $l : C(N) \to \mathbb{R}_+$ as a length measure on the continuum of points. For two points $p, p' \in C(N)$, we denote by $\Pi(p, p') \subseteq 2^{C(N)}$ and $\Pi^*(p, p') \subseteq \Pi(p, p')$ the set of paths and shortest paths, respectively, connecting p and p'. Any path $\pi \in \Pi(p, p')$ is indifferently treated as a continuum in $C(N)$, then $l_\pi := l(\pi)$ is the length of π. The distance between p and p', $d(p, p')$, is the length of a shortest path connecting them:

$$d(p, p') := \min\{l_\pi : \pi \in \Pi(p, p')\} = l_{\pi^*} \text{ for any } \pi^* \in \Pi^*(p, p').$$

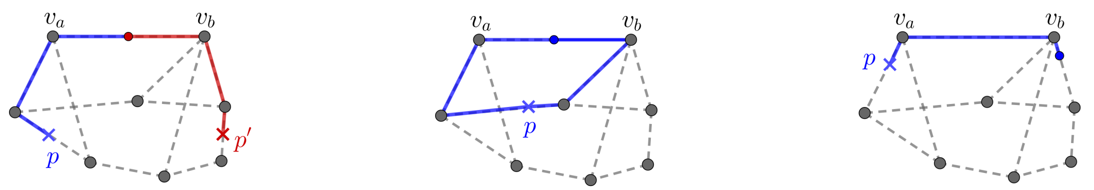
In particular, if p and p' belong to the same edge, we denote by $l(p, p')$ the length of the unique path in that edge connecting them. We work under the following assumption:

Assumption 2.1. $\delta \geq l_e$ for all $e \in E$.

If Assumption 2.1 did not hold, we could consider a set $I \subset 2^{C(N)}$, that would contain, for each $e = (v_a, v_b) \in E$, the following continuum sets of points (segments):

- If $\delta \geq l_e$, $e \in I$;
- If $\delta < l_e$, let $n := \lceil \frac{l_e}{\delta} \rceil + 1$. We define $v_1 := v_a$, $v_n := v_b$ and $v_2, \ldots, v_{n-1} \in e$ such that $l(v_a, v_i) = (i-1)\frac{l_e}{n-1}$ for $i = 2, \ldots, n-1$. Then, $(v_i, v_{i+1}) \in I$ for all $i = 1, \ldots, n-1$.

We consider $N' = (V', E' = I)$, where V' contains the end points of I. The new network N' satisfies that $\delta \geq l_e$ for all $e \in E'$, and it is isomorphic to N with respect to the length function. Indeed, since N' is obtained by subdividing edges in N, $C(N) = C(N')$ and a set of points $\delta-$covers N if and only it $\delta-$covers N'. Therefore, Assumption 2.1 always holds after the network N is transformed into N' (via a preprocessing step).



(a) By two points (b) By one point, through both ends (c) By one point, trough one end

Figure 3: Covering of an edge $e = (v_a, v_b) \in E$

As a consequence of Assumption 2.1, if $e = (v_a, v_b) \in E$ and $p \in C(N)$, then p can δ-cover either the whole e, or a continuous part of e containing either v_a or v_b. Following this idea, the next proposition states a characterization of the δ-covers of a network.

Proposition 2.1. Let $\mathcal{P} = \{p_i\}_{p_i \in C(N)}$ be a finite set of points in $C(N)$. An edge $e = (v_a, v_b) \in E$ is δ-covered by $\mathcal{P}$ if and only if either there exists $p \in \mathcal{P} \cap e$ or

$$\max\{\delta - \min_{p \in \mathcal{P}} d(v_a, p), 0\} + \max\{\delta - \min_{p \in \mathcal{P}} d(v_b, p), 0\} \geq l_e. \tag{2}$$

The set $\mathcal{P}$ is a $\delta-$cover of N if and only if for each $e \in E$, either there exists $p \in \mathcal{P} \cap e$ or (2) is satisfied.

Proof. If there exists $p \in \mathcal{P} \cap e$, then e is $\delta-$covered by p due to Assumption 2.1. Otherwise, for each $i \in \{a, b\}$, let us consider $p_i^* \in \mathcal{P}$ such that $d(v_i, p_i^*) = \min_{p \in \mathcal{P}} d(v_i, p)$ and let $\pi_i^* \in \Pi^*(v_i, p_i^*)$ be a shortest path between v_i and p_i^*, i.e. $l_{\pi_i^*} = d(v_i, p_i^*)$. Condition (2) can be rewritten as follows:

$$\max\{\delta - l_{\pi_a^*}, 0\} + \max\{\delta - l_{\pi_b^*}, 0\} \geq l_e.$$

Note that $\max\{\delta - l_{\pi_i^*}, 0\}$ represents the maximum length that can be $\delta-$covered by $\mathcal{P}$ (specifically, from p_i^*) after passing through v_i. Since the path(s) that δ-cover e must contain v_a and/or v_b, the edge is covered if and only if these "maximum lengths" for v_a and v_b add up to more than l_e.

□

Figure 3 illustrates Proposition 2.1. It shows three ways of covering the same edge $e = (v_a, v_b) \in E$ for a given network. The edges of the network are depicted with dashed lines, while the different paths through which e is covered are delimited with continuous bold traces. Facility locations are marked with the symbol 'x'. Figure 3a depicts two facilities located at p, p' that cover two portions of the edge, which contain v_a and v_b respectively. In this case, the min functions inside (2) are attained respectively at p and p'. In the middle, Figure 3b shows a single location p that covers e through two different paths, which traverse v_a and v_b respectively. These paths form a cycle that contains p and e. In this case, the two min operations inside (2) are attained at the same point, p. Finally, Figure 3c illustrates the case in which a single facility located at p covers e through one of its end nodes, v_a. Here, one of the max operators in (2) is equal to zero (p is further from v_b than δ).

The characterization in Proposition 2.1 is based on the individual covering of every edge in the network. When considering possible locations to cover a fixed edge, we can restrict ourselves to its surroundings within the radius δ. Delimiting those parts of the network that could "contribute" to cover a particular edge or node reduces the search space. The sets in the following definition delimit the edges and nodes of $C(N)$ that can contribute to covering a particular node of the network.

Definition 2.1. For each $v \in V$, the *potential covers* of v are the candidate facility locations to cover v:

$$\begin{aligned}
&\mathcal{E}(v) := \{e' = (v'_a, v'_b) \in E : \ d(v, v'_i) \leq \delta \text{ for some } i \in \{a, b\}\} \\
&\mathcal{V}(v) := \{v' \in V : \ d(v, v') \leq \delta\} \\
&\mathcal{F}(v) := \mathcal{E}(v) \cup \mathcal{V}(v).
\end{aligned}$$

Clearly, v is not reachable within the radius δ for any facility installed outside $\mathcal{F}(v)$. Regarding the covering of edges, an edge incident to v could be covered by some of the facilities in $\mathcal{F}(v)$. The following definition serves to delimit the parts of the network where, if a facility is installed, it will completely cover a particular edge.

Definition 2.2. For each $e = (v_a, v_b) \in E$, the *complete covers* of e are the candidate facility locations that can completely cover e:

$$\begin{aligned}
&\mathcal{E}_{\mathrm{c}}(e) := \{e' \in E : \forall p' \in e', \forall p \in e, \ d(p, p') \leq \delta\} \\
&\mathcal{V}_{\mathrm{c}}(e) := \{v' \in V : \forall p \in e, \ d(p, v') \leq \delta\} \\
&\mathcal{F}_{\mathrm{c}}(e) := \mathcal{E}_{\mathrm{c}}(e) \cup \mathcal{V}_{\mathrm{c}}(e).
\end{aligned} \tag{3}$$

If a facility is placed at $\mathcal{F}_{\mathrm{c}}(e)$ (either at a node in $\mathcal{V}_{\mathrm{c}}(e)$ or at a point on an edge belonging to $\mathcal{E}_{\mathrm{c}}(e)$), we can immediately conclude that e is $\delta-$covered. Note that any facility placed at e' can completely cover e if and only if any facility placed at e can completely cover e'. That is, $\mathcal{E}_{\mathrm{c}}$ is symmetric over E. On the other hand, it is obvious that $e \in \mathcal{E}_{\mathrm{c}}(e)$, and $v_a, v_b \in \mathcal{V}_{\mathrm{c}}(e)$, for all $e = (v_a, v_b) \in E$. The following definition identifies those candidate facility locations in the potential covers of a node v that cannot completely cover any incident edges to v.

Definition 2.3. We define the following sets for each $v \in V$:

$$\begin{aligned}
&\mathcal{E}_{\mathrm{p}}(v) := \{e' \in \mathcal{E}(v) : \ \exists e \in E(v), \ e' \notin \mathcal{E}_{\mathrm{c}}(e)\} \\
&\mathcal{V}_{\mathrm{p}}(v) := \{v' \in \mathcal{V}(v) : \ \exists e \in E(v), \ v' \notin \mathcal{V}_{\mathrm{c}}(e)\} \\
&\mathcal{F}_{\mathrm{p}}(v) := \mathcal{V}_{\mathrm{p}}(v) \cup \mathcal{E}_{\mathrm{p}}(v).
\end{aligned} \tag{4}$$

We call these sets the *partial covers* of $E(v)$, where $E(v) := \{e \in E : \ v \in e\}$ is the set of incident edges to v.

The set $\mathcal{F}_{\mathrm{p}}(v)$ contains those candidate locations that can contribute to partially covering some of the edges in $E(v)$. Note that, if a facility is placed at $\mathcal{F}(v) \setminus \mathcal{F}_{\mathrm{p}}(v)$, then this facility completely covers $E(v)$.

Definitions 2.1, 2.2, and 2.3 provide us with a refined covering condition. Indeed, the following proposition is a consequence of Proposition 2.1 and the aforementioned definitions, and will be used to characterize coverings in the MILP formulation presented in Section 3.

Proposition 2.2. A discrete set $\mathcal{P}$ of points in $C(N)$ is a $\delta-$cover of N if and only if, for each $e = (v_a, v_b) \in E$, either $\mathcal{P} \cap \mathcal{F}_{\mathrm{c}}(e) \neq \emptyset$ or

$$\sum_{i \in \{a,b\}} \min \left\{ \max\left\{0, \delta - \min_{p \in \mathcal{P} \cap \mathcal{F}_{\mathrm{p}}(v_i)} d(v_i, p)\right\}, l_e \right\} \geq l_e, \tag{5}$$

Moreover,

$$\min_{p\in\mathcal{P}\cap\mathcal{F}_{\rm p}(v_i)} d(v_i,p) = \min\left\{\min_{v'\in\mathcal{P}\cap\mathcal{V}_{\rm p}(v_i)} d(v_i,v'),\ \min_{p\in\mathcal{P}\cap(\mathcal{E}_{\rm p}(v_i)\setminus V)} d(v_i,p)\right\},\ \text{for } i=a,b.$$

Proof. For $i \in \{a, b\}$, the equality $\mathcal{F}(v_i) = \mathcal{F}_{\rm p}(v_i) \cup \mathcal{F}_{\rm c}(e)$ holds by definition, which gives the new necessary and sufficient covering condition.

For the second statement of the proposition, we have $\mathcal{F}_{\rm p}(v_i) = \mathcal{V}_{\rm p}(v_i) \cup \mathcal{E}_{\rm p}(v_i)$ from Definition 2.3. Then, if suffices to see that we can take $p \in \mathcal{P} \cap (\mathcal{E}_{\rm p}(v_i) \setminus V)$ in the second inner min operator of the right-hand side instead of $p \in \mathcal{P} \cap \mathcal{E}_{\rm p}(v_i)$. Let $e' = (v'_a, v'_b) \in \mathcal{E}_{\rm p}(v_i)$. We prove that the end nodes of e' can be excluded from the second inner min operator. Let $i' \in \{a, b\}$, we denote $\bar{i}' = b$ if $i' = a$ and $\bar{i}' = a'$ if $i' = b$. First, by definition, $d(v_i, v'_{i'}) \le \delta$ for some $i' \in \{a, b\}$. Therefore, $v'_{i'} \in \mathcal{V}_{\rm p}(v_i)$, and we can exclude it from the second inner min operator (this node is already considered by the first inner min operator). We consider now the other end node of e'. If $d(v_i, v'_{\bar{i}'}) \le \delta$ then, similarly, $v'_{\bar{i}'}$ can be excluded from the second inner min. Otherwise, we know that the outer min is not attained at $v'_{\bar{i}'}$, as $d(v_i, v'_{i'}) \le \delta < d(v_i, v'_{\bar{i}'})$, thus $v'_{\bar{i}'}$ can be disregarded. □

Remark 2.1. The covering conditions described both in Propositions 2.1 and 2.2 would be applicable if only a subset of E, $J \subseteq E$, is to be covered. Indeed, these covering conditions are based on the individual coverage of the edges, so it would be sufficient to apply them just to the edges in J. As a consequence, our methods, including the MILP formulation and algorithms presented in the next sections, apply to this more general version of the CSCP$_\delta$.

In order to exploit the newly defined potential, complete, and partial covers in our formulation, from a practical viewpoint, we need to have some characterizations that can operate in a computer. Definition 2.1, which introduces potential covers, satisfy this requirement. Indeed, it just depends on distances between pairs of nodes, which we can easily calculate. Conversely, Definition 2.2 presents complete covers with a condition that must be satisfied by "the infinitely many points of an edge", which is not directly computable. Finally, the elements in the partial covers defined by Definition 2.3 can be easily calculated once both potential and complete covers are known.

In the following, we focus on characterizing the complete covers, which will be useful for our MILP formulation and tailored algorithms, (see forthcoming Sections 3 and 5). To begin with, we note that Proposition 2.1 already gives us a characterization of the nodes in $\mathcal{V}_{\rm c}(e)$. Indeed, it is easy to observe that, for a given $e \in E$, $v \in \mathcal{V}_{\rm c}(e)$ if and only if $\mathcal{P} := \{v\}$ δ-covers e. We then focus on the sets $\mathcal{E}_{\rm c}(e)$. On the one hand, it is clear that, for every edge $e = (v_a, v_b) \in E$,

$$\mathcal{E}_{\rm c}(e) \subseteq \mathcal{E}(v_a) \cap \mathcal{E}(v_b).$$

Moreover, if we define $\mathcal{E}_{\rm c}(v) \subseteq \mathcal{E}(v)$ as follows, we have a tighter set containing $\mathcal{E}_{\rm c}(e)$.

Definition 2.4. The edges that can completely cover a node $v \in V$ are:

$$\mathcal{E}_{\rm c}(v) := \{e' \in E : \forall p' \in e',\ d(v, p') \le \delta\}.$$

It is clear that $\mathcal{E}_{\rm c}(e) \subseteq \mathcal{E}_{\rm c}(v_a) \cap \mathcal{E}_{\rm c}(v_b)$, for all $e = (v_a, v_b) \in E$. We recall that Definition 2.4 is somewhat the inverse of Definition 2.2. That is, $e' \in \mathcal{E}_{\rm c}(v)$ if and only if $v \in \mathcal{V}_{\rm c}(e')$. We present a set of intermediate statements in Definition 2.5, Lemma 2.1, and Lemma 2.2, which allow us to describe the edges in the complete cover sets $\mathcal{E}_c(e)$ as main result in Proposition 2.3.

Definition 2.5. Let $v \in V$ be a node and $e' = (v'_a, v'_b) \in E \setminus E(v)$ be an edge. For all $q \in [0, l_{e'}]$, we define the following functions:

$$d_v(q) := \min\{d(v, v'_a) + q, d(v, v'_b) + l_{e'} - q\}$$
$$r_v(q) := \max\{\delta - d_v(q), 0\}.$$

The function $d_v(q)$ represents the distance between v and a point $p' \in e'$ such that $q = l(v'_a, p')$, where $l(v'_a, p')$ measures the length of the continuum $(v'_a, p') \subseteq e'$. We define the constant $Q_v := (d(v, v'_b) + l_{e'} - d(v, v'_a))/2$, which satisfies the following equation:

$$d(v, v'_a) + Q_v = d(v, v'_b) + l_{e'} - Q_v.$$

That is, the inner terms in the minimization that defines $d_v(q)$ coincide for $q = Q_v$. Informally, Q_v is the "equilibrium" coordinate on e', for which the distance to v is the same if we go through v'_a or v'_b. Indeed, since $|d(v, v'_b) - d(v, v'_a)| \leq l_{e'}$, it follows that $0 \leq Q_v \leq l_{e'}$. Note that $d_v(q)$, $r_v(q)$, and Q_v depend also on the edge e'. However, the associated edge e' will be always clear from the context, and we avoid including it in our notation for the sake of readability.

Lemma 2.1. Let $e = (v_a, v_b)$. An edge $e' \in \mathcal{E}_c(v_a) \cap \mathcal{E}_c(v_b)$ is in $\mathcal{E}_c(e)$ if and only if $r_{v_a}(q) + r_{v_b}(q) \geq l_e$ for all $q \in [0, l_{e'}]$.

Proof. $e' \in \mathcal{E}_c(e)$ if and only e is $\delta-$covered by any point $p' \in e'$. Take $\mathcal{P} = \{p'\}$ in Proposition 2.1, e is $\delta-$covered by $\mathcal{P}$, if and only if

$$\max\{\delta - d(v_a, p'), 0\} + \max\{\delta - d(v_b, p'), 0\} \geq l_e$$

holds for all $p' \in e'$. Let $q = l(v'_a, p')$, $q \in [0, l_{e'}]$, be the measure of the sub-edge $(v'_a, p') \subseteq e'$. Then, the observation that $d_{v_i}(q) = d(v_i, p')$ for $i = \{a, b\}$ completes the proof. ☐

We present the following lemma without proof. The lemma is a direct consequence of Definition 2.5.

Lemma 2.2. Let $v \in V$ and $e' = (v'_a, v'_b) \in E \setminus E(v)$. The function $d_v(q)$ is increasing when $q \in [0, Q_v]$, and it decreases for $q \in [Q_v, l_{e'}]$. Moreover, $d_v(q)$ admits the following piece-wise linear representation:

$$d_v(q) = \begin{cases} d(v, v'_a) + q & \text{if } q \leq Q_v, \\ d(v, v'_b) + l_{e'} - q & \text{if } q \geq Q_v. \end{cases}$$

On the other hand, we define

$$\underline{Q_v} := \max\{q \in [0, Q_v] : \ d(v, v'_a) + q \leq \delta\} \text{ and}$$
$$\overline{Q_v} := \min\{q \in [Q_v, l_{e'}] : \ d(v, v'_b) + l_{e'} - q \leq \delta\}.$$

Note that, if $e' \in \mathcal{E}_c(v)$, then $\underline{Q_v}$ and $\overline{Q_v}$ are properly defined by the above equations (the min and max operators are not over empty sets). Moreover, $\underline{Q_v}$ and $\overline{Q_v}$ admit the following analytical expressions:

$$\underline{Q_v} = \min\{Q_v, \delta - d(v, v_a')\}; \qquad \overline{Q_v} = \max\{Q_v, d(v, v_b') + l_{e'} - \delta\}.$$

Then, if $e' \in \mathcal{E}_\mathrm{c}(v)$, $r_v(q)$ admits the following piece-wise linear representation:

$$r_v(q) = \begin{cases} \delta - (d(v, v_a') + q) & \text{if } q \leq \underline{Q_v}, \\ 0 & \text{if } \underline{Q_v} \leq q \leq \overline{Q_v}, \\ \delta - (d(v, v_b') + l_{e'} - q) & \text{if } q \geq \overline{Q_v}. \end{cases}$$

Proposition 2.3. Let $e = (v_a, v_b)$. An edge $e' \in \mathcal{E}_\mathrm{c}(v_a) \cap \mathcal{E}_\mathrm{c}(v_b)$ is in $\mathcal{E}_\mathrm{c}(e)$ if and only if

$$r(q) \geq l_e \quad \text{for all} \quad q = \underline{Q}, \overline{Q},$$

where $r(q) := r_{v_a}(q) + r_{v_b}(q)$, $\underline{Q} := \max\{\underline{Q_{v_a}}, \underline{Q_{v_b}}\}$, and $\overline{Q} := \min\{\overline{Q_{v_a}}, \overline{Q_{v_b}}\}$.

Proof. From Lemma 2.1, $e' \in \mathcal{E}_\mathrm{c}(e)$ if and only if $\min_{q \in [0, l_{e'}]} r(q) \geq l_e$. Due to Lemma 2.2, the minimum argument must be some of the breakpoints in the piece-wise linear description of $r_v(q)$. Then, it suffices to check

$$\min\{r(q) : \ q \in \{0, \underline{Q_{v_a}}, \underline{Q_{v_b}}, \overline{Q_{v_a}}, \overline{Q_{v_b}}, l_{e'}\}\} \geq l_e.$$

Since $e' \in \mathcal{E}_\mathrm{c}(v_a) \cap \mathcal{E}_\mathrm{c}(v_b)$, $r_{v_i}(0) \geq l_e$ and $r_{v_i}(l_{e'}) \geq l_e$ always holds for all $i \in \{a, b\}$. It suffices thus to check the following condition

$$\min\{r(q) : \ q \in \{\underline{Q_{v_a}}, \underline{Q_{v_b}}, \overline{Q_{v_a}}, \overline{Q_{v_b}}\}\} \geq l_e.$$

Let $m = \arg\max_{i \in \{a,b\}}\{\underline{Q_{v_i}}\}$ and $k = \arg\min_{i \in \{a,b\}}\{\underline{Q_{v_i}}\}$, $k \neq m$. Then, $r_{v_m}(q)$ has constant derivative equal to -1 for $q \in [0, \underline{Q}]$. On the other hand, $r_{v_k}(q)$ has derivative in $\{-1, 0, 1\}$ for $q \in [0, l_{e'}]$. Since $r = r_{v_m} + r_{v_k}$, r has non-positive derivative for $q \in [0, \underline{Q}]$, i.e. it is non-increasing. Consequently, it must be that $r(\underline{Q}) \leq r(\underline{Q_{v_i}})$ $(i \in \{a, b\})$. Similarly, it must be that $r(\overline{Q}) \leq r(\overline{Q_{v_i}})$ $(i \in \{a, b\})$. □

3. MILP formulation

We have the following observations of an optimal $\delta-$cover, which can be yielded from the previous section (a similar observation was proven in [16]):

i) Each edge $e \in E$ has at most two facilities (due to Assumption 2.1);

ii) If there are two facilities in the edge e, we can assume without loss of generality that they are located at end nodes v_a, v_b (this follows from i) and Assumption 2.1).

Therefore, the set of candidate facilities of a δ-cover is in one-to-one correspondence to the sets E and V. For each $v \in V$, there is one candidate facility (fixed location); and for each $e \in E$, there is another one (location within the interior of e, $e \setminus \{v_a, v_b\}$). Then, the discrete set $\mathcal{F} := E \cup V$ will be used to index the candidate facilities. In our MILP formulation, there are two decisions associated with each $f \in \mathcal{F}$. One is to decide if a facility is installed at f. The second is only necessary for those facilities installed at the interior of edges, and consists in determining their locations within the corresponding edges.

To represent the first of the above decisions, we define the following binary variables, which we call the *placement variables*:

$$y_f = 1 \text{ iff a facility is installed at } f, \quad \text{for all } f \in \mathcal{F}.$$

We identify the set of installed facilities with $\mathcal{F}_1 = \{f \in \mathcal{F} : y_f = 1\}$. To represent the second of the above decisions, we define the following continuous variables, which we name the *coordinate variables*:

$$q_e = \begin{cases} l(v_a, p) & \text{if } y_e = 1 \text{ and a facility is installed at } p \in e \\ 0 & \text{otherwise} \end{cases} \qquad \text{for all } e = (v_a, v_b) \in E.$$

We use v' and e' to denote nodes and edges where facilities are installed, and use v and e to denote nodes and edges to be covered, respectively. We refer to v_a, v_b as the end nodes of $e = (v_a, v_b) \in E$; given $e' = (v'_a, v'_b) \in E$, we refer to v'_a, v'_b as its end nodes.

We can now present our MILP formulation of the CSCP$_\delta$. We use the necessary and sufficient condition of $\delta-$covering in Proposition 2.2, and the second result in this proposition regarding the distance function. Other than the placement and coordinate variables, some additional variables are used, which we present next.

$$\begin{array}{lll}
w_e \in \{0,1\} & = 1 \text{ if } \mathcal{F}_\mathrm{c}(e) \cap \mathcal{F}_1 \neq \emptyset & e \in E; \\
r_v \in [0, M_v] & \leq \max\{0, \delta - \min\limits_{f \in \mathcal{F}_\mathrm{p}(v) \cap \mathcal{F}_1} \{d(v,p) : \text{ a facility is installed at } p \in f\}\} & v \in V; \\
x_v \in \{0,1\} & = 1 \text{ if, for all } e \in E(v) \text{, a facility is installed at } \mathcal{F}_\mathrm{c}(e) & v \in V; \\
z_{vv'} \in \{0,1\} & = 1 \text{ if, for all } e = (u,v) \in E(v) \text{ s.t. } w_e = 0,\ \max\{0, \delta - d(v,v')\} + r_u \geq l_e & v \in V,\ v' \in \mathcal{V}_\mathrm{p}(v); \\
z_{ve'i'} \in \{0,1\} & = 1 \text{ if, for all } e = (u,v) \in E(v) \text{ s.t. } w_e = 0,\ \max\{0, \delta - \tau_{ve'i'}(q_{e'})\} + r_u \geq l_e & v \in V,\ (e', i') \in \mathcal{EI}_\mathrm{p}(v),
\end{array}$$

where

$$\tau_{ve'i'}(q) := d(v, v_{i'}) + \mathbb{1}_{i'=a} q + \mathbb{1}_{i'=b}(l_{e'} - q), \quad \text{and} \quad \mathcal{EI}_\mathrm{p}(v) := \{(e' = (v'_a, v'_b), i') \in \mathcal{E}_\mathrm{p}(v) \times \{a,b\} : d(v, v'_{i'}) \leq \delta\}.$$

We sometimes refer to r_v as the "residual cover" at node v, since it represents the maximum remaining length that can be covered after reaching v from "a sufficiently close" facility. These variables satisfy:

$$r_v \leq r_v(q_e), \text{ where } q_e = l(v_a, p) \text{ for } p = \arg\min\{d(v,p) : \text{ a facility is installed at } p\},\ p \in e = (v_a, v_b) \in E.$$

In an optimal solution, it can be $r_v = r_v(q_e)$ for an edge $e \in E$ as stated above. In this case, r_v is the maximum remaining length that can be covered after reaching v from the closest facility. However, we do not impose this equality in our formulation, since it is enough for guaranteeing the coverage of $e = (v_a, v_b) \in E$ that the sum of the residuals $r_{v_a} + r_{v_b}$ exceeds l_e (see Proposition 2.2). That is, if l_e is already exceeded by $r_{v_a} + r_{v_b}$ for some $r_{v_i} < r_{v_i}(q_e)$, then the coverage condition of Proposition 2.2 will hold.

Our formulation of the CSCP_δ reads as follows:

$$
\begin{aligned}
& \min \sum_{f \in \mathcal{F}} y_f && && \text{(6a)}\\
& \text{s.t. } w_e \geq y_f && e \in E, f \in \mathcal{F}_\text{c}(e) && \text{(6b)}\\
& w_e \leq \sum_{f \in \mathcal{F}_\text{c}(e)} y_f && e \in E && \text{(6c)}\\
& x_v \geq 1 - \sum_{e \in E(v)} (1 - w_e) && v \in V && \text{(6d)}\\
& x_v \leq w_e && v \in V, e \in E(v) && \text{(6e)}\\
& y_{v'_{i'}} + y_{e'} \leq 1 && e' \in E, i' \in \{a, b\} && \text{(6f)}\\
& q_{e'} \leq l_{e'} y_{e'} && e' \in E && \text{(6g)}\\
& l_e(1 - w_e) \leq r_{v_a} + r_{v_b} && e \in E && \text{(6h)}\\
& x_v + \sum_{v' \in \mathcal{V}_\text{p}(v)} z_{vv'} + \sum_{(e', i') \in \mathcal{EI}_\text{p}(v)} z_{ve'i'} = 1 && v \in V && \text{(6i)}\\
& z_{vv'} \leq y_{v'} && v \in V, v' \in \mathcal{V}_\text{p}(v) && \text{(6j)}\\
& z_{ve'i'} \leq y_{e'} && v \in V, (e', i') \in \mathcal{EI}_\text{p}(v) && \text{(6k)}\\
& r_v \leq M_v(1 - x_v) && v \in V && \text{(6l)}\\
& r_v \leq M_{vv'}(1 - z_{vv'}) + \delta - d(v, v') && v \in V, v' \in \mathcal{V}_\text{p}(v) && \text{(6m)}\\
& r_v \leq M_{ve'i'}(1 - z_{ve'i'}) + \delta - \tau_{ve'i'}(q_{e'}) && v \in V, (e', i') \in \mathcal{EI}_\text{p}(v) && \text{(6n)}\\
& y_f, w_e \in \{0, 1\} && f \in \mathcal{F}, e \in E && \text{(6o)}\\
& x_v, z_{vv'}, z_{ve'i'} \in \{0, 1\} && v \in V, v' \in \mathcal{V}_\text{p}(v), (e', i') \in \mathcal{EI}_\text{p}(v) && \text{(6p)}\\
& q_{e'}, r_v \geq 0 && e' \in E, v \in V. && \text{(6q)}
\end{aligned}
$$

Constraints (6b) and (6c) model the logic or constraint $w_e = \vee_{f \in \mathcal{F}_\text{c}(e)} y_f$. Constraints (6d) and (6e) enforce the logic constraint $x_v = \wedge_{e \in E(v)} w_e$, that is, x_v is the product of the w_e variables such that $e \in E(v)$. Constraints (6f) prevent two facilities in a solution from being installed respectively at the interior of an edge and one of their end nodes. Constraints (6g) bound the coordinate variables with the corresponding edge length, and set them to zero if no facility is located at its interior. The covering condition in Proposition 2.2 is enforced by (6h). If $w_e = 1$, then the condition is satisfied (e is covered by $\mathcal{F}_\text{c}(e)$). Otherwise, the inequality (5) of the proposition has to be satisfied. The rest of the constraints of the model (6i)-(6n), together with variables r, x, q, and z, aim at modeling (5). To begin with, (6i) impose that, for each $v \in V$, one of the following statements holds:

i) All incident edges to v, $e \in E(v)$, are completely covered by facilities placed at their complete covers, $\mathcal{F}_\text{c}(e)$ ($w_e = 1$ for all $e \in E(v)$, $x_v = 1$).

ii) A sufficiently close facility to v is installed at $v' \in \mathcal{V}_\text{p}(v)$ ($z_{vv'} = 1$), that is,

$$\max\{0, \delta - d(v, v')\} + r_u \geq l_e \qquad \forall u \in V \text{ s.t. } (u, v) = e \in E \text{ and } w_e = 0;$$

iii) A sufficiently close facility to v is installed at $e' \in \mathcal{E}_\text{p}(v)$ and v is reached through $v'_{i'}$ of e' ($z_{ve'i'} = 1$), that is,

$$\max\{0, \delta - \tau_{ve'i'}(q_{e'})\} + r_u \geq l_e \qquad \forall u \in V \text{ s.t. } (u,v) = e \in E \text{ and } w_e = 0.$$

If the case i) above holds, then the covering condition in (6h) is satisfied for all $e \in E(v)$, regardless of the value of the residual cover variables. Otherwise, suppose that $x_v = 0$ and $w_e = 0$ for some $e \in E(v)$. In this case, the corresponding constraint (6h) is "active", that is, the inequality (5) of Proposition 2.2 has to be satisfied for e. Since $x_v = 0$, constraints (6i) impose that there is a facility among those installed at $\mathcal{F}_{\mathrm{p}}(v)$ that is sufficiently close one to v. This facility is the one bounding the residual variables r_v (see constraints (6m)-(6n)), which represent the terms in the left-hand side of (5). Constraints (6j) (resp. (6k)) ensure that $z_{vv'}$ (resp. $z_{ve'i'}$) can be one only if facility is installed at v' (resp. e'). Due to (6i), for every fixed node $v \in V$, at most one of the constraints in (6l)-(6n) will be active. If $x_v = 1$, (6l) enforces $r_v = 0$. Indeed, all the covering conditions (6h) are "inactive" and r_v is not needed to guarantee the coverage of any $e \in E(v)$. Otherwise, if $x_v = 0$, (6l) reads $r_v \leq M_v$, where M_v is a big-enough constant that does not restrict the value of the residual. Finally, constraints (6m)-(6n) bound r_v by $\delta - d(v,p) \geq 0$ for a sufficiently close facility to v installed at p, when $x_v = 0$. The constants $M_{vv'}$ and $M_{ve'i'}$ are assumed to be big enough so that the constraints in (6m)-(6n) do not add anything to the model if $z_{vv'}$ or $z_{ve'i'}$ are zero, respectively. For instance, $M_v = M_{vv'} = \delta$ and $M_{ve'i'} = \delta + l_{e'}$ are valid values for these constants (we recall Assumption 2.1). Section 4 presents refined values of these big-Ms.

We observe that the number of variables and constraints in (6) can be reduced. Namely, for each $v \in V$ and $e' = (v'_a, v'_b) \in \mathcal{E}_{\mathrm{p}}(v)$, if $d(v, v'_a) + l_{e'} \leq d(v, v'_b)$ then $d(v,p) = d(v, v'_a) + l(v'_a, p)$ for every $p \in e'$. Similarly, if $d(v, v'_b) + l_{e'} \leq d(v, v'_a)$ then $d(v,p) = d(v, v'_b) + l(v'_b, p)$ always holds for all $p \in e'$. For such nodes and candidate facilities, we do not need both variables, $z_{ve'a}$ and $z_{ve'b}$, and corresponding constraints in (6n) (we know beforehand that one of these constraints would never be active if a facility is located at e'). Therefore, $\mathcal{EI}_{\mathrm{p}}(v)$ would only contain one of the pairs (e', a) or (e', b).

3.1. Comparative insights with respect to an existing MILP

To the best of our knowledge, the only existing MILP for the CSCP$_\delta$ was proposed in [16]. The authors used a similar observation to ours with respect to optimal δ-covers. They noted that every edge contains at most one facility. Indeed, in their setting, if two facilities are located at both end-nodes of an edge $e = (v_a, v_b)$ one of them is considered to be "owned" by an adjacent edge, $e' \in E(v_a) \cup E(v_b)$ (by optimality, neither v_a nor v_b is a leaf). Their location variables are indexed then by E. However, this approach has symmetric issues. Indeed, there are many symmetric equivalent solutions of an optimal δ-cover, since there exist exponentially many combinations of the edges "owning" the facilities that are located at nodes. A second main difference between MILP (6) and the MILP in [16] is that the latter uses binary variables to identify the two edges containing the facilities that cover a given edge. On the one hand, this yields variables and constraints of $\mathcal{O}(|E|^3)$. On the other hand, multiple equivalent solutions arise when the edge in question can be covered by a single facility, as the authors commented themselves. Finally, the covering constraints in both formulations actually correspond to the same characterization of δ-cover, but modeled in a slightly different way. Namely, the authors of [16] defined the "residual covers" for each edge (where a facility might be placed) and node of the network. The interested readers might consult [16] and the MILP therein, which we do not reproduce here for the sake of concision. Nonetheless, Table 1 shows a comparative summary of the two formulations, based on the number of variables and constraints. This summary considers an upper bound on the size of MILP (6). That is, we take $\mathcal{F}_{\mathrm{c}}(e) = \mathcal{F}$, $\mathcal{V}_{\mathrm{p}}(v) = V$, and $\mathcal{E}_{\mathrm{p}}(v) = E$ for all $v \in V$ and $e \in E$—however, this

	Variables		Constraints
	Binaries	Continuous	
MILP (6)	$\lvert V\rvert^2 + 2(\lvert V\rvert\lvert E\rvert + \lvert V\rvert + \lvert E\rvert)$	$\lvert V\rvert + \lvert E\rvert$	$\lvert E\rvert^2 + \lvert V\rvert^2 + 5\lvert E\rvert\lvert V\rvert + 7\lvert E\rvert + 3\lvert V\rvert$
MILP in [16]	$\lvert E\rvert^3 + 3\lvert V\rvert\lvert E\rvert + \lvert E\rvert$	$3\lvert V\rvert\lvert E\rvert + \lvert E\rvert$	$3\lvert E\rvert^3 + 8\lvert E\rvert\lvert V\rvert + \lvert E\rvert$

Table 1: Comparative summary on MILP formulations for the CSCP$_\delta$

would never be the case, as the partial and complete covers are complementary. On the other hand, Table 1 considers the MILP in [16] with $J = E$ (the set of edges to be covered).

4. Strengthening

In this section, we analyze modifications of the MILP (6) that can yield a tighter linear relaxation of this formulation. Namely, we tight our big-M constraints (6l)-(6n) by devising small constants M_v, $M_{vv'}$, $M_{ve'i'}$, $\delta_{vv'}$, and $\delta_{ve'i'}$. We also present several families of valid inequalities.

4.1. Constants tightening

From the MILP formulation, it is easy to yield the following observation. For $v \in V$, it suffices for a facility $f \in \mathcal{F}_{\mathrm{p}}(v)$ to contribute to the residual cover r_v at most $U_v := \max_{e \in E(v)} l_e$. Indeed, r_v aims at ensuring that the inequality (5) of Proposition 2.2 is satisfied for all $e \in E(v)$. We define

$$\begin{aligned} \delta_{vv'} &:= \min\{U_v + d(v,v'), \delta\}, && \text{for } v' \in \mathcal{V}_{\mathrm{p}}(v), \text{ and} \\ \delta_{ve'i'} &:= \min\{U_v + \max_{q \in [0, l_{e'}]} \tau_{ve'i'}(q), \delta\} = \min\{U_v + d(v, v'_{i'}) + l_{e'}, \delta\}, && \text{for } (e' = (v'_a, v'_b), i') \in \mathcal{EI}_{\mathrm{p}}(v). \end{aligned}$$

Since U_v is a valid upper bound for the residual cover variable r_v, the big-Ms in the constraints (6m) and (6n) should guarantee that

$$\begin{aligned} M_{vv'} + \delta_{vv'} - d(v,v') &\geq U_v, \\ M_{ve'i'} + \min_{q \in l_{e'}} (\delta_{ve'i'} - \tau_{ve'i'}(q)) &\geq U_v. \end{aligned}$$

Taking the minimums of the above big-Ms, we can now tighten the big-M constants of the MILP (6) as follows:

$$\begin{aligned} M_v &:= U_v \\ M_{vv'} &:= U_v - (\delta_{vv'} - d(v,v')) = \max\{0, U_v + d(v,v') - \delta\} \\ M_{ve'i'} &:= U_v - \min_{q \in l_{e'}} (\delta_{ve'i'} - \tau_{ve'i'}(q)) = U_v - \delta_{ve'i'} + \max_{q \in l_{e'}} \tau_{ve'i'}(q) \\ &= U_v - \delta_{ve'i'} + d(v_i, v'_{i'}) + l_{e'} = \max\{0, U_v + d(v, v'_{i'}) + l_{e'} - \delta\}, \end{aligned}$$

where the last equations in the definition of $M_{vv'}$ and $M_{ve'i'}$ follow from the definition of $\delta_{vv'}$ and $\delta_{ve'i'}$, respectively.

(a) A facility is located at v

(b) Solution with two facilities

Figure 4: Illustration of valid inequalities (9)

Consequently, the constraints (6m) and (6n) should be replaced by:

$$r_v \leq M_{vv'}(1 - z_{vv'}) + \delta_{vv'} - d(v, v') \qquad v \in V, v' \in \mathcal{V}_{\text{p}}(v) \tag{7a}$$

$$r_v \leq M_{ve'i'}(1 - z_{ve'i'}) + \delta_{ve'i'} - \tau_{ve'i'}(q_{e'}) \qquad v \in V, (e', i') \in \mathcal{EI}_{\text{p}}(v). \tag{7b}$$

4.2. Valid inequalities

"Leafs" inequalities

If a node $v \in V$ has degree one, we can assume without loss of generality that no facility is located at v nor at its incident edge. Indeed, an equivalent δ-cover could be built by just moving such a facility to the unique neighbor of v in N. More than valid inequalities, the following are valid variable elimination:

$$y_v = 0;\; y_e = 0 \quad \forall v \in V \;\text{ s.t. }\; deg(v) = 1,\; e \in E(v). \tag{8}$$

"Adjacent edges" inequalities

Consider a node $v \in V$ of degree two. If there is a facility at v, then no facility is placed at the edges incident to v (we recall the model constraints (6f)). Otherwise, we can assume that at most one facility is placed at these edges in an optimal solution, which can be enforced by the following valid inequalities:

$$y_e + y_{e'} + y_v \leq 1 \quad \forall e, e' \in E,\; e \neq e',\; s.t.\; e \cap e' = v \text{ and } deg(v) = 2. \tag{9}$$

Figure 4 illustrates the above inequalities. Figure 4a shows the case in which a facility is located at v. Otherwise, if two facilities are placed at e and e' respectively, we can build an equivalent solution by moving one of these facilities to the end node of the corresponding edge that is not v, as depicted in Figure 4b. We recall that the last statement holds due to our Assumption 2.1.

"Neighborhood" inequalities

Let us now consider a node $v \in V$ and suppose that there are several facilities placed at different edges in $E(v)$ in a feasible solution. Take $e^* \in E(v)$ containing a facility f^* such that $d(f^*, v) = \min\{d(f, v) : f \text{ is installed at } e \in E(v)\}$. The following proposition gives an equivalent feasible solution where the facilities at the edges $e \in E(v)$ such that $e \neq e^*$ are moved to the nodes.

Proposition 4.1. For any feasible solution $\hat{y}$ with several facilities placed at edges in $E(v)$, the following solution y is feasible and $\sum_{f \in \mathcal{F}} y_f \leq \sum_{f \in \mathcal{F}} \hat{y}_f$:

- $y_u = 1$ for all $u \in V$ such that $e = (u, v) \in E$, $e \neq e^*$, and $\hat{y}_e = 1$;
- $y_e = 0$ for all $e \in E(v)$ such that $e \neq e^*$, and $\hat{y}_e = 1$;
- $y_f = \hat{y}_f$ otherwise.

(a) A facility is located at v (b) Solution with several facilities

Figure 5: Illustration of valid inequalities (10)

Proof. We denote by $N(v)$ the set of vertices adjacent to v. Consider the change of facilities from $\hat{y}$ to y. The facilities in the edges $E(v) \setminus \{e\}$ are 'pushed' to the vertices $N(v)$. For $u \in N(v)$, if there already exists a facility at u, and there is another facility 'pushed' to u, then these two facilities merge and they are accounted as one facility in y. Hence, the number of facilities of solution y is at most that of solution $\hat{y}$.

. The proof then consists in showing that y is feasible. We will show that all edges are covered. Let us consider $e \in E$. If e was covered in $\hat{y}$ by facilities not placed at edges in $E(v)$ then it is still covered by these facilities in y. Suppose then that a facility placed at $e' \in E(v)$ with $e \neq e^*$ was covering e (or part of e) in solution $\hat{y}$, and let $e' = (u, v)$. We distinguish two cases. First, if the facility at e' was partially covering e through node u, then it clearly covers at least the same part of e in the new solution y (where the facility is moved to u). Otherwise, suppose the facility at e' was partially covering e through node v. In this case, the facility at e^* covers at least the same part of e (it is closer to v). Since this facility remains unchanged in the new solution, we can guarantee that e is still covered. □

As a consequence of Proposition 4.1, the following inequalities are valid:

$$\sum_{e \in E(v)} y_e \leq 1 - y_v \qquad \forall v \in V. \tag{10}$$

Figure 5 illustrates the valid inequalities (10). In particular, Figure 5b illustrates the equivalent solution given in Proposition 4.1. It is easy to observe that these new inequalities are a generalization of inequalities (9). Moreover, constraints (10) dominate the model constraints (6f)— and are fewer.

5. Network processing

The network processing algorithm analyzes N to compute the parameters and sets needed to construct the MILP model (6), which we recall next:

1. $\mathcal{V}_{\mathrm{c}}(e)$, $\mathcal{E}_{\mathrm{c}}(e)$ for all edges $e \in E$;

2. $\mathcal{V}_{\mathrm{p}}(v), \mathcal{E}_{\mathrm{p}}(v), \mathcal{EI}_{\mathrm{p}}(v)$ for all nodes $v \in V$;

3. $d(v, v')$ for all pairs of nodes $v, v' \in V$ such that $d(v, v') \leq \delta$.

The above data is computed by Algorithms 1, 2 and 3. Algorithms 1 and 2 contain auxiliary functions, which are called within the main Algorithm 3. Algorithm 1 computes the sets $\mathcal{E}(v)$ and $\mathcal{V}(v)$ (which are not directly used in the MILP but necessary to obtain $\mathcal{E}_{\mathrm{p}}(v)$ and $\mathcal{V}_{\mathrm{p}}(v)$), and the distances $d(v, v')$ for all $v, v' \in V$ such that $d(v, v') \leq \delta$. Algorithm 1 also computes the sets $\mathcal{E}_{\mathrm{c}}(v)$, which will serve as intermediate sets to finally obtain $\mathcal{E}_{\mathrm{c}}(e)$ in Algorithm 3. The main task in Algorithm 3 is to compute the sets $\mathcal{V}_{\mathrm{c}}(e)$ and $\mathcal{E}_{\mathrm{c}}(e)$. To that aim, this algorithm calls both Algorithm 1 and the procedure "mutual" described in Algorithm 2. Once $\mathcal{V}_{\mathrm{c}}(e)$ and $\mathcal{E}_{\mathrm{c}}(e)$ are known, the computation of $\mathcal{V}_{\mathrm{p}}(v)$ and $\mathcal{E}_{\mathrm{p}}(v)$ in Algorithm 3 easily follows by definition.

In the following, we present Algorithm 1, which defines the function "nodeCover(N, δ, s)". This function, for each source node $s \in V$, outputs: $\mathcal{E}_{\mathrm{c}}(s)$, $\mathcal{E}(s)$, $\mathcal{V}(s)$, and $d(s, v)$ for all $v \in V$ such that $d(s, v) \leq \delta$ (otherwise the algorithm outputs $d(s, v) = +\infty$). The algorithm starts with empty sets $\mathcal{E}_{\mathrm{c}}(s), \mathcal{E}(s), U(s), \mathcal{V}(s)$, where $U(s)$ is used for intermediate calculations. The set Q denotes nodes whose shortest path (and distance) to s are unknown, and it is initialized to V. In course of the algorithm, Q decreases, while $\mathcal{V}(s)$ increases. In Lines 7-11, the distance $d(s, v)$ and predecessor values $\mathrm{prev}_s(v)$ are initialized, for all $v \in V$. The while loop is an adaptation of the classic Dijkstra algorithm. Line 14 selects the node u with the shortest distance to s among all unprocessed nodes, and removes it from Q. If $d(s, u) > \delta$, then none of the remaining nodes in Q are reachable from s, and the search is pruned. Otherwise, the neighbors of u that are sill in Q are inspected. For each $v \in Q \cap E(u)$, the edge (u, v) is first added to $\mathcal{E}(s)$. Then, the algorithm computes the length ℓ of a path from s to v that traverses u. If $\ell < d(s, v)$, then the distance and the predecessor for node v are updated in Lines 24- 25. In addition, if $\ell < \delta$, node v and edge (u, v) are added to $\mathcal{E}_{\mathrm{c}}(s)$ and $\mathcal{V}(s)$ in Lines 27 and (28), respectively. Otherwise, the edge e is added to the undetermined set $U(s)$. Whether this edge belongs or not to the complete cover set $\mathcal{E}_{\mathrm{c}}(s)$ is decided later on in the algorithm. Namely, edges $e = (v_a, v_b) \in U(s)$ are processed in Lines 35-39: if e can be jointly $\delta-$covered by s from two sides, then e is added to the complete cover $\mathcal{E}_{\mathrm{c}}(s)$.

Algorithm 2 describes the procedure "mutual", which determines, given $e = (v_a, v_b) \in E$ and a candidate edge for the complete cover $e' \in \mathcal{E}_{\mathrm{c}}(v_a) \cap \mathcal{E}_{\mathrm{c}}(v_b)$, whether $e' \in \mathcal{E}_{\mathrm{c}}(e)$. This algorithm is based on Proposition 2.3 in Section 2.

Network processing Algorithm 3 computes all the sets that are needed by the MILP formulation. The algorithm starts with empty sets $\mathcal{E}_{\mathrm{c}}(e), \mathcal{V}_{\mathrm{c}}(e), \mathcal{E}_{\mathrm{p}}(v), \mathcal{V}_{\mathrm{p}}(v)$, for $e \in E$ and $v \in V$. In Line 3, the algorithm loops through all nodes $v \in V$ and computes the function "nodeCover(N, δ, v)", storing its output. Then, the algorithm calculates the sets $\mathcal{V}_{\mathrm{c}}(e)$ for $e \in E$, by applying the symmetric relation between these sets and the sets $\mathcal{E}_{\mathrm{c}}(v)$ from "nodeCover(N, δ, v)". After that, in Line 10, the algorithm loops through all edges $e = (v_a, v_b) \in E$. It checks whether there is an edge $e' \in \mathcal{E}_{\mathrm{c}}(v_a) \cap \mathcal{E}_{\mathrm{c}}(v_b)$ such that $e' \in \mathcal{E}_{\mathrm{c}}(e)$ (equivalently, $e \in \mathcal{E}_{\mathrm{c}}(e')$) by calling the procedure "mutual". Since $e' \in \mathcal{E}_{\mathrm{c}}(e)$ if and only if $e \in \mathcal{E}_{\mathrm{c}}(e')$, the loop only runs over pairs such that $e < e'$ (we assume an total order on the elements of E). The loop starting in line 18, iterates on each node $v \in V$ and looks for $v' \in \mathcal{V}(v)$ such that there exists an $e \in E(v)$ but $e \notin \mathcal{E}_{\mathrm{c}}(v')$. The nodes v' found are added to $\mathcal{V}_{\mathrm{p}}(v)$. Finally, the loop in line 28 also iterates on $v \in V$, and looks for $e' = (v'_a, v'_b) \in \mathcal{E}(v)$ such that there exists $e \in E(v)$ but $e \notin \mathcal{E}_{\mathrm{c}}(e')$. Each edge found is added to $\mathcal{E}_{\mathrm{p}}(v)$, and, right after that, the set $\mathcal{EI}_{\mathrm{p}}(v)$ may be updated after checking the dominance rule described at the end of Section 3. We have the following complexity result for Algorithm 3.

Proposition 5.1. Let D be an upper bound on the degree of the nodes of a connected network $N = (V, E)$. The time complexity of the network processing Algorithm 3 is $\mathcal{O}(|E|^2 + |V||E|(D + \log |V|))$.

Proof. We first analyze the time complexity of the procedure nodeCover described in Algorithm 1. The main **while** loop is a modification of the Dijkstra algorithm, and it can be implemented with time complexity $\mathcal{O}((|E| + |V|) \log |V|)$, see [19]]. Therefore, the overall time complexity of nodeCover is also $\mathcal{O}((|E| + |V|) \log |V|)$.

The network processing Algorithm 3 has two **for** loops over V and E, which calculate the complete cover sets. Then, the next two **for** loops are over V and calculate the partial cover sets. For the each iteration of the first loop, there is a call to nodeCover, and hence its time complexity is $\mathcal{O}(|V|((|E| + |V|) \log |V|))$. For the each iteration of the second **for** loop, there is a call to mutual (which has constant complexity), and hence the time complexity is $\mathcal{O}(|E|^2)$. For each of the remaining **for** loops, there is an intermediate loop over

V (resp. E), and finally an inner loop over incident edges (bounded by D). The overall time complexity of these two loops is then $\mathcal{O}(|V||E|D)$ (since N is connected). Therefore, the time complexity of Algorithm 3 is $\mathcal{O}(|V|((|E|+|V|)\log|V|)+|E|^2+|V||E|D)$, or, equivalently, $\mathcal{O}(|V||E|\log|V|+|E|^2+|V||E|D)$. □

Algorithm 1: single node δ-cover algorithm: nodeCover

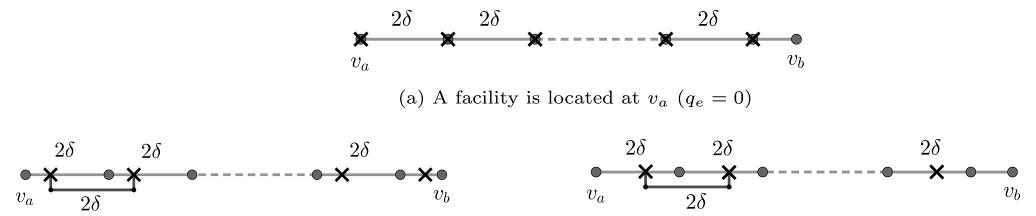
```
Input: Network N = (V, E, ||), cover range δ > 0, a source s ∈ V;
Output: E_c(s), E(s), V(s), d(s,v) for all v ∈ V (returns d(s,v) = +∞ if d(s,v) > δ);
Initialize set Q ← V;
Initialize sets E_c(s) ← ∅, E(s) ← ∅, U(s) ← ∅;
Initialize set V(s) ← ∅;
for each node v ∈ V do
    d(s,v) ← +∞ ;                                  ▷ Unknown distance from s to v
    prev_s(v) ← {∅} ;                              ▷ Unknown predecessor of v
end
d(s,s) ← 0;
add s to V(s);
while Q is not empty do
    u ← arg min_{v∈Q} d(s,v);
    remove u from Q ;                              ▷ Take the closest node u and remove it from Q
    if d(s,u) > δ then
        d(s,v) ← +∞ for all v ∈ Q ; ▷ End of Dijkstra (all nodes in Q are outside the covering radius)
        break
    end
    for each v ∈ Q s.t. v ∈ E(u) do
        e ← (u,v);
        add e to E(s) ;                            ▷ Edge e is in the potential cover set of s
        ℓ ← d(s,u) + l_e ;                         ▷ Path from s to v that traverses u
        if ℓ < d(s,v) then
            d(s,v) ← ℓ ;                           ▷ Update the distance to v
            prev_s(v) ← u ;                        ▷ Update the predecessor of v
            if ℓ ≤ δ then
                add e to E_c(s) ;                  ▷ Edge e is in the complete cover set of s
                add v to V(s) ;                    ▷ Node v is in the potential cover set of s
            else
                add e to U(s) ;                    ▷ Undetermined edge
            end
        end
    end
end
for each edge e = (v_a, v_b) in U(s) do
    if v_a ∈ V(s) and v_b ∈ V(s) and δ − d(s,v_a) + δ − d(s,v_b) ≥ l_e then
        add e to E_c(s) ;                          ▷ Edge e is completely covered
    end
end
```

Algorithm 2: Edge mutual cover algorithm: mutual

Input: Edges $e = (v_a, v_b), e' = (v'_a, v'_b)$ such that $e' \in \mathcal{E}_\mathrm{c}(v_a)$ and $e \in \mathcal{E}_\mathrm{c}(v_b)$.
Output: Boolean value indicating whether $e' \in \mathcal{E}_\mathrm{c}(e)$.
for $i \in \{a, b\}$ **do**
    $Q_{v_i} \leftarrow \frac{d(v_i, v'_b) + l_{e'} - d(v_i, v'_a)}{2}$;
    $\underline{Q_{v_i}} \leftarrow \max\{\min\{Q_v, \delta - d(v, v'_a)\}, 0\}$;
    $\overline{Q_{v_i}} \leftarrow \min\{\max\{Q_v, d(v, v'_b) + l_{e'} - \delta\}, l_{e'}\}$;
end
$\underline{Q} \leftarrow \max\{\underline{Q_{v_a}}, \underline{Q_{v_b}}\}$;
$\overline{Q} \leftarrow \min\{\overline{Q_{v_a}}, \overline{Q_{v_b}}\}$;
for $i \in \{a, b\}$ **do**
    for $q \in \{\underline{Q}, \overline{Q}\}$ **do**
        if $q \leq \underline{Q_{v_i}}$ **then**
            $r_{v_i}(q) = \delta - (d(v_i, v'_a) + q)$;
        else if $\underline{Q_{v_i}} \leq q \leq \overline{Q_{v_i}}$ **then**
            $r_{v_i}(q) = 0$;
        else
            $r_{v_i}(q) = \delta - (d(v_i, v'_b) + l_{e'} - q)$;
        end
    end
end
if $r_{v_a}(\underline{Q}) + r_{v_b}(\underline{Q}) \geq e$ AND $r_{v_a}(\overline{Q}) + r_{v_b}(\overline{Q}) \geq e$ **then**
    return TRUE;
else
    return FALSE;
end

Algorithm 3: Network processing algorithm

Input: Network $N = (V, E, ||)$ and cover range $\delta > 0$;
Output: $\mathcal{E}_{\rm c}(e)$, $\mathcal{V}_{\rm c}(e)$, $\mathcal{E}_{\rm p}(v)$, $\mathcal{V}_{\rm p}(v)$, $\mathcal{EI}_{\rm p}(v)$, for all $e \in E$ and $v \in V$, and distance function d;
for *each node* $v \in V$; ▷ Computation of node complete covers $\mathcal{V}_{\rm c}(e)$
do
$\mathcal{E}_{\rm c}(v), \mathcal{E}(v), \mathcal{V}(v), d(v, \cdot) \leftarrow \text{nodeCover}(N, \delta, v)$;
for *each edge* $e \in \mathcal{E}_{\rm c}(v)$ **do**
add v to $\mathcal{V}_{\rm c}(e)$;
end
end
for *each edge* $e = (v_a, v_b) \in E$; ▷ Computation of edge complete covers $\mathcal{E}_{\rm c}(e)$
do
for *each edge* $e' = (v'_a, v'_b) \in E,\ e < e'$*, such that* $e' \in \mathcal{E}_{\rm c}(v_a) \cap \mathcal{E}_{\rm c}(v_b)$ **do**
if *mutual*(e, e', d) **then**
add e' to $\mathcal{E}_{\rm c}(e)$;
add e to $\mathcal{E}_{\rm c}(e')$;
end
end
for *each node* $v \in V$; ▷ Computation of node partial covers $\mathcal{V}_{\rm p}(v)$
do
for *each node* $v' \in \mathcal{V}(v)$ **do**
for *all* $e \in E(v)$ **do**
if $e \notin \mathcal{E}_{\rm c}(v')$ **then**
add v' to $\mathcal{V}_{\rm p}(v)$;
break
end
end
end
for *each node* $v \in V$; ▷ Computation of edge partial covers $\mathcal{E}_{\rm p}(v)$ and $\mathcal{EI}_{\rm p}(v)$
do
for *each edge* $e' = (v'_a, v'_b) \in \mathcal{E}(v)$ **do**
for *all* $e \in E(v)$ **do**
if $e' \notin \mathcal{E}_{\rm c}(e)$ **then**
add e' to $\mathcal{E}_{\rm p}(v)$;
if $d(v, v'_a) \leq \delta$ AND $d(v, v'_a) \leq d(v, v'_b) + l_{e'}$ **then**
add (e', a) to $\mathcal{EI}_{\rm p}(v)$;
if $d(v, v'_b) \leq \delta$ AND $d(v, v'_b) \leq d(v, v'_a) + l_{e'}$ **then**
add (e', b) to $\mathcal{EI}_{\rm p}(v)$;
break
end
end
end



(a) A facility is located at v_a ($q_e = 0$)

(b) A facility is located at the tail ($0 < q_e \leq \hat{l}_e$)

(c) No facility is located at the tail ($\hat{l}_e < q_e \leq 2\delta$)

Figure 6: Illustration of Proposition 6.1 for a long edge e

6. A reduced formulation for networks with long edges

The MILP (6) assumes $l_e \leq \delta$ for all $e \in E$, however, in many real-world networks, some edges' lengths are greater than the covering radius. One approach is to transform such networks by subdividing their edges as suggested in Section 2. After that, we obtain a new network with maximum edge length at most δ, and the previous MILP (6) can be applied. However, this strategy increases the number of edges and nodes of the network, and thus the number of variables and constraints of the model by a nonlinear factor. In this section, we present an alternative approach to tackle networks with edges' lengths greater than the covering radius. Instead of transforming the network, this approach directly treats long edges in the formulation by using specific sets of constraints and variables. We highlight that the approach is also applicable to "long paths". That is, if there is a path in the network with all their middle-nodes having degree two, we can represent it by a single edge of length equal to the total length of the path. Indeed, the CSCP_δ does not change after this transformation.

The main idea of the reduced formulation is to assume a predefined covering of those edges that are long enough. Such covering consists in placing facilities every 2δ distance units on the long edge. Let us consider $e \in E$ such that $l_e > 2\delta$. An edge satisfying this condition is called a *long edge*. We denote by $\hat{l}_e := l_e - 2\delta\lfloor l_e/(2\delta)\rfloor$ the length of the last piece of e after dividing it into pieces of measure 2δ. We call $\hat{l}_e$ the *tail* of e. The following proposition guarantees the correctness of the reduced formulation.

Proposition 6.1. Let N be an undirected network, $e = (v_a, v_b) \in E$ be a long edge, and $\mathcal{P}'$ be a feasible δ-cover of N. Define $\mathcal{P}$ with $p \in \mathcal{P}$ for all $p \in \mathcal{P}' \setminus e$. Let $p_e \in \mathcal{P}' \cap e$ be such that $l(v_a, p_e) = \min_{p' \in \mathcal{P}' \cap e} l(v_a, p')$, and let $q_e := l(v_a, p_e)$ (here q_e represents a length, although it will also be a variable of the reduced MILP that we introduce afterwards). Note that $q_e \in [0, 2\delta]$ (otherwise, $\mathcal{P}'$ would not be a δ-cover). The set $\mathcal{P}$ can be completed in such a way that it δ-covers N and $|\mathcal{P}| \leq |\mathcal{P}'|$, as follows:

- $p_e \in \mathcal{P}$;
- $p \in \mathcal{P}$ for all $p \in e$, $p > p_e$, such that $l(p_e, p) = 2\delta \cdot k$, for some $k \in \mathbb{N}$;
- If $\exists p' \in \mathcal{P}' \cap e$ such that $d(p_e, p') > 2\delta\lfloor l(p_e, v_b)/(2\delta)\rfloor$ then $v_b \in \mathcal{P}$.

Moreover, $\lfloor l_e/(2\delta)\rfloor \leq |\mathcal{P} \cap e| \leq \lfloor l_e/(2\delta)\rfloor + 2$. In particular,

(i) If $0 \leq q_e \leq \hat{l}_e$, then $|\mathcal{P} \cap e| = \lfloor l_e/(2\delta)\rfloor + 1$ if $v_b \notin \mathcal{P}$, $|\mathcal{P} \cap e| = \lfloor l_e/(2\delta)\rfloor + 2$ otherwise.

(ii) If $\hat{l}_e < q_e \leq 2\delta$, then $|\mathcal{P} \cap e| = \lfloor l_e/(2\delta)\rfloor$ if $v_b \notin \mathcal{P}$, $|\mathcal{P} \cap e| = \lfloor l_e/(2\delta)\rfloor + 1$ otherwise.

Proof. It is easy to observe that $|\mathcal{P}| \leq |\mathcal{P}'|$. First, since facilities are placed every 2δ distance on e, the original covering $\mathcal{P}'$ cannot contain fewer facilities than $\mathcal{P}$. Since the rest of the facilities are just taken from $\mathcal{P}'$,

$|\mathcal{P}| \leq |\mathcal{P}'|$. On the other hand, $\mathcal{P}$ has to δ-cover N for the same reason. That is, the facilities of $\mathcal{P}$ that were not in $\mathcal{P}'$ cover at least as much as the ones originally in $\mathcal{P}'$. The last part of the proposition easily follows from construction, and is illustrated by Figure 6. □

In the following, we present our reduced formulation, which is an adaptation of MILP (6). We treat long edges specifically to subdivide them into smaller edges, improving the scalability of our approach. Edges $e \in E$ such that $\delta < l_e \leq 2\delta$ are subdivided into two subedges of length smaller than δ. Therefore, we assume that, for every $e \in E$, either $l_e \leq \delta$ or $l_e > 2\delta$. In the former case, all the constraints and variables of the model remain unchanged. In the latter, we introduce new variables and constraints to the model, while dropping some of the constraints originally in (6). The objective function also needs adaptation. We introduce all these modifications next.

Let $e = (v_a, v_b) \in E$ be a long edge and, for any feasible solution, let q_e be as in Proposition 6.1. That is, when e is a long edge, we use the former variable q_e of MILP (6) to represent the position of the left-most facility on e with respect to v_a. The placement variables y_{v_a}, y_{v_b}, and y_e will be used as well, with slightly different meanings to those in (6), as we will explain later on. We introduce an indicator variable $u_e \in \{0, 1\}$ to distinguish between two possible ranges in the domain of q_e. If $0 \leq q_e \leq \hat{l}_e$, then $u_e = 0$; otherwise $\hat{l}_e \leq q_e \leq 2\delta$ and $u_e = 1$. This can be modelled with the following constraints:

$$\begin{aligned} q_e &\leq \hat{l}_e(1 - u_e) + 2\delta u_e, \\ q_e &\geq \hat{l}_e u_e. \end{aligned} \tag{11}$$

From Proposition 6.1, there is a transition in the number of facilities on e when u_e changes from 0 to 1. Let us denote by $L \subseteq E$ the set of long edges of the network. The objective function of the reduced MILP reads as follows

$$\sum_{f \in \mathcal{F} \setminus L} y_f + \sum_{e \in L} \Big(\Big\lceil \frac{l_e}{2\delta} \Big\rceil - u_e \Big). \tag{12}$$

Note that the coefficients on the last term in the objective already account for the facilities installed at v_a for each $e = (v_a, v_b) \in L$, while they do not do so for v_b. This will condition the values of the placement variables in an optimal solution, namely, $y_{v_a} = 0$ for all $e = (v_a, v_b) \in L$. The facilities installed at these nodes will be tracked by the variables q_e, namely, if $q_e = 0$ then a facility would be installed at v_a. To complete the modeling of the CSCP$_\delta$, we need to ensure that the covering of the long edge fits into the covering of the rest of the network. Namely, some parts of the network might be covered by facilities placed on e, and part of e (namely its tail or the portion between v_a and p_e of Proposition 6.1) could be covered by facilities placed outside e.

We focus first on the case $0 \leq q_e < \hat{l}_e$. We need to ensure that both the segment (v_a, p_e) and the tail of e are covered. For the tail, we know that there is a facility at a distance $\hat{l}_e - q_e$ from v_b. This facility covers a length δ of the remaining fragment on e on its right-hand side, which has length equal to $\hat{l}_e - q_e$. The rest of such fragment should be covered, which can be imposed by the following constraint:

$$r_{v_b} \geq \hat{l}_e - q_e - \delta \iff r_{v_b} + q_e + \delta \geq \hat{l}_e \qquad \forall e \in L \text{ s.t. } u_e = 0. \tag{13}$$

To ensure the covering of the segment (v_a, p_e), we have:

$$r_{v_a} + \delta \geq q_e \qquad \forall e \in L \text{ s.t. } u_e = 0.$$

Let us consider now the case $\hat{l}_e \leq q_e \leq 2\delta$. There is a facility installed at a distance $2\delta - q_e + \hat{l}_e$ from v_b. Then, to ensure that the tail of e is covered, we need to cover the fragment between this facility and v_b. Since the facility already covers a length δ on this fragment, the following constraint enforces the covering of the tail:

$$r_{v_b} \geq 2\delta - q_e + \hat{l}_e - \delta \iff r_{v_b} + q_e - \delta \geq \hat{l}_e \qquad \forall e \in L \text{ s.t. } u_e = 1. \tag{14}$$

To ensure the covering of the segment (v_a, p_e), we have the same equation as before:

$$r_{v_a} + \delta \geq q_e \qquad \forall e \in L \text{ s.t. } u_e = 1.$$

In summary, the following constraints ensure that the edge e is fully covered:

$$r_{v_a} + \delta \geq q_e \quad \forall e \in L, \tag{15}$$

$$r_{v_b} + q_e - (2u_e - 1)\delta \geq \hat{l}_e \quad \forall e \in L. \tag{16}$$

Constraint (16) gathers (13) and (14) in a single constraint. The reduced MILP model is as follows (we avoid an extended writing of the model for the sake of conciseness):

$$\begin{aligned}
\min \quad & (12) & \\
\text{s.t.} \quad & (6b), (6c), (6h) & e \notin L \\
& (6f), (6g), (6k) & e' \notin L \\
& (6d), (6e), (6i), (6j), (6l), (6m), (6n), (6o), (6p), (6q) & \\
& (11), (15), (16) & \\
& y_e = 1 & e \in L \qquad (17)\\
& w_e = 0 & e \in L, \qquad (18)
\end{aligned}$$

where, if $e \in L$, the term $\tau_{ve_l i}(q_e)$ in (6n) is replaced by $d(v, v_i) + \mathbb{1}_{i=a} q_e + \mathbb{1}_{i=b}(2\delta u_e + \hat{l}_e - q_e)$. We enforce (17) because e always contains a facility if $e \in L$. On the other hand, we need to include constraints (18) to guarantee that the variables r_{v_a} and r_{v_b} can take positive values. Indeed, if $w_e = 1$ for $e \in L$, it may happen that $x_{v_a} = 1$ or $x_{v_b} = 1$, which will will imply, respectively, $r_{v_a} = 0$ or $r_{v_b} = 0$ due to (6l). We compute the complete and partial cover sets in the same way as for the original MILP model. Note that there is no edge or node that can completely cover e if $e \in L$.

The following theorem is on the scalability of the reduced MILP above.

Theorem 6.1. Given a network $N = (V, E, l)$, the maximum number of variables and constraints of the reduced MILP model only depends on V and E.

Proof. The number of constraints and variables of the reduced model does not grow with the edges' lengths, except for a constant factor of 2 for those edges $e \in E$ such that $\delta < l_e \leq 2\delta$. □

7. Computational results

In this section, we present the computational experiments testing the existing and proposed formulations and strengthening techniques for CSCP_δ and its discrete variant (facilities on nodes).

7.1. Experiment Setup

In this section, we describe the setup of the experiments including the benchmarks, development environment, implementation of algorithms and solution statistics. The computational results and source code are publicly released on our project website: https://github.com/lidingxu/cflg/, where we provide a bash file to reproduce the experiments in Linux systems. Those benchmarks that we generated for this study, or that were publicly available already, are also available at the repository.

Benchmarks. We use three different benchmarking sets: two come from the literature, and the other has been generated synthetically. For every instance, we set the coverage radius δ equal to the average of the edge lengths. We describe these benchmarks next.

`Kgroup.` It consists of 23 prize-collecting Steiner tree problem instances from [20], and the benchmark includes the graphs and edge lengths of these instances. These random geometric instances are designed to have a local structure somewhat similar to street maps. Nodes correspond to random points in the unit square. The number of nodes ranges from 22 to 241. There is an edge between two nodes if their distance is no more than a prescribed threshold which depends on the number of nodes, and the length of an edge is the Euclidean distance between the two points. It is divided into two sets, `Kgroup_A` and `Kgroup_B`. The first one consists of 12 small instances with up to 45 nodes, and the second one consists of 11 large instances with up to 241 nodes.

`City.` It consists of real data of 9 street networks for some German cities, and it was first used in [13]. The number of nodes ranges from 132 to 771. The length of each edge is the length of the underlying street segment.

`Random.` It consists of 24 random networks instances generated via Erdős-Rényi binomial method with the package “Networkx” (see [21]). A network is constructed by connecting nodes randomly. Each edge is included with a predefined uniform probability p. The number of nodes, n, is in $\{10, 15, 20, 25, 30, 40\}$. For each n, we generate random graphs with different adjacency probabilities, namely $p \in \{0.1, 0.2, 0.3, 0.4\}$. Furthermore, we split these instances into two benchmarks: `Random_A` and `Random_B`. `Random_A` contains instances with $n \in \{10, 15, 20\}$. `Random_B` contains instances with $n \in \{25, 30.40\}$.

Problem preprocessing. Graphs of instances are modified in a problem preprocessing step to be amenable to MILP models.

Given an original network of each instance, in the first preprocessing step, we delete any degree-two node and concatenate its adjacent edges to a new edge, as long as the deletion does not yield a self-loop. Such a node can be treated as an interior point of the new edge. We refer to the preprocessed network without any such degree-two node as the degree-two-free network.

Even after the first preprocessing step, the degree-two-free network may not correspond to the actual problem network to solve, since we may subdivide the degree-two-free network for the non-reduced model to guarantee that $\delta > \max_{e \in E} |e|$. We refer to the preprocessed network after the second preprocessing step as the subdivided network, which is degree-two-free and satisfies $\delta > \max_{e \in E} |e|$. Therefore, the size (number of nodes and edges) of a subdivided network depends on δ.

Development environment. The experiments are conducted on a computer with Intel Core i7-6700K CPU @ 4.00GHZ and 16GB main memory. JuMP [22] is a modeling language for mathematical optimization embedded in Julia. We use JuMP to implement our models and interact with MILP solvers. Specifically, we use ILOG CPLEX 20.1 to solve our models. Alternatively, the implementation allows users to switch easily to other solvers (e.g. Gurobi and GLPK).

CPLEX's parameters are set as their defaults, except that we disable its parallelism and set the MIP absolute gap to 1 (due to the integeral objective). The experiments are partitioned into jobs. Every job calls CPLEX to solve an instance, and this job is handled by one process of the multi-core CPU. To safeguard against a potential mutual slowdown of parallel processes, we run only one job per core at a time, and we use at most three processes in parallelism. The time limit of each job is set to 1800 CPU seconds.

Model implementation. We implement six models based on different combinations of formulations and settings. The first five models address CSCP_δ, while the last model solves its discrete restriction, i.e. the variant in which facilities must be placed at nodes. These models are as follows.

`EF`. This model implements the model from [16] for CSCP_δ. This formulation only uses edges to model facility locations, and the authors do not consider the complete and partial cover sets to delimit the size of the model. This model assumes $\delta > \max_{e \in E} |e|$, and it reads the subdivided graph.

`F0`. This model implements a basic formulation that is a simplification of the model (6). It does not use the complete and partial cover information nor any of the strengthening techniques in Section 4. Hence, it does not call the network processing algorithm nodeCover. This model assumes $\delta > \max_{e \in E} |e|$, and it reads the subdivided graph. More precisely, the constraints (6b)-(6e) related to complete covers are removed, the complete cover variables w are fixed to 0; for each $v \in V$, the partial cover sets $\mathcal{E}_\text{p}(v)$, $\mathcal{EI}_\text{p}(v)$ are solely set, respectively, as E and $E \times \{a, b\}$, and consequently, $M_v = \delta$, $M_{vv'} = r(N)$ for $v' \in \mathcal{E}_\text{p}(v)$, $M_{ve'i'} = r(N) + |e'|$ for $(e', i') \in \mathcal{EI}_\text{p}(v)$ are trivial valid bound constants, where $r(N) := \max_{v, v' \in N} d(v, v')$ is the radius of the problem network N.

`F`. This model implements the complete formulation (6) for CSCP_δ, it does use the complete and partial cover information, and hence it calls the network processing algorithm nodeCover. It does not use the strengthening techniques in Section 4. This model assumes $\delta > \max_{e \in E} |e|$, and it reads the subdivided network as well. For each $v \in V$, due to the delimited partial cover set, $M_v = \delta$, $M_{vv'} = \delta$ for $v' \in \mathcal{E}_\text{p}(v)$, $M_{ve'i'} = \delta + |e'|$ for $(e', i') \in \mathcal{EI}_\text{p}(v)$ are valid bound constants.

`SF`. This model strengthens `F` by using the techniques described in Section 4. More precisely, the big-M constants are reduced as Section 4.1; the "Leafs" inequalities are used to fix variables; and the "Neighborhood" inequalities are implemented as model constraints which replace (6f).

`RF`. This model implements the reduced formulation from Section 6. It only requires $\delta < 2\max_{e \in E} |e|$. Given a degree-two-free network, it models the long edge specifically as the description Section 6, and it subdivides the edges with length greater than δ and smaller than 2δ into two sub-edges.

`SFD`. Any solution of the discrete restriction of CSCP_δ—where facilities can only be placed at nodes— is a feasible solution of CSCP_δ. We name this discrete restriction by the discrete facility covering problem. This model solves the discrete facility covering problem, which solely sets $y_e = 0$ for all $e \in E$ in `SF` model.

The above models are summarized in Table 2. Both `EF` and `F0` consider that any two points in the network can possibly cover each other, and do not utilize the complete and partial cover information. They have been already compared in Section 3.1, and hence `F0` should have fewer variables and constraints than `EF`. For these models, we are interested in the dual gaps obtained after the models are solved within the time limit.

Performance metrics and statistical tests. We describe the performance metrics and the ways to compute their statistics. These statistics will be used to evaluate the model performance.

Let $\underline{v}$ be a dual lower bound and $\overline{v}$ be a primal upper bound obtained after solving some of the models

Model	Problem	Set delimitation	Strengthening	Long edge	Size	Input network	Comment
`EF`	$CSCP_\delta$	No	No	No	Very large	Subdivided network	From [16]
`F0`	$CSCP_\delta$	No	No	No	Large	Subdivided network	The simple model
`F`	$CSCP_\delta$	Yes	No	No	Medium	Subdivided network	The complete model
`SF`	$CSCP_\delta$	Yes	Yes	No	Medium	Subdivided network	The strengthened model
`RF`	$CSCP_\delta$	Yes	Yes	Yes	Small	Degree-two-free network	The reduced model
`SFD`	discrete facility covering problem	Yes	Yes	No	Very Small	Subdivided network	The discrete model

Table 2: Model summary

described above, the relative dual gap in percentage is defined as:

$$\sigma := \frac{\overline{v} - \underline{v}}{\overline{v}} \times 100.$$

A smaller relative dual gap indicates better primal and dual behaviour of the model.

Let n_{sd} be the number of nodes of the subdivided network of that instance, note that a trivial primal solution is the set of the nodes of the subdivided network (for which edge length is at most δ). Therefore, to normalize the primal solution value, we define the relative primal bound in percentage

$$v_r := \frac{\overline{v}}{n_{sd}} \times 100.$$

If $v_r < 100\%$, then the model finds a solution better than the trivial one.

In order to evaluate model performance, we compute shifted geometric means (SGMs) of performance metrics, which provides a measure for relative differences. This avoids statistics being dominated by outliers with large absolute values as is the case for the arithmetic mean. The SGM also avoids an over representation of results with small absolute values. The SGM of values $v_1, ..., v_M \geq 0$ with shift $s \geq 0$ is defined as

$$\left(\prod_{i=1}^{M}(v_i + s)\right)^{1/M} - s.$$

We say an instance is affected by a model, if solving this model finds a feasible solution; the instance is solved by this model, if solving this model finds an optimal solution. If an instance is unaffected, usually the model is too large to be read into the MILP solver.

We record the following performance metrics of each instance for each model, and compute the benchmark-wise SGMs:

1. t: the total running time in CPU seconds, with a shifted value set to 1 second;
2. $\sigma(\%)$: the relative dual gap in percentage, with a shifted value set to 1%;
3. $v_r(\%)$: the relative primal bound in percentage, with a shifted value set to 1%.

For an unaffected instance, we set by default $t = 1800$, $\sigma = 100$ and $v_r = 100$. Note that the time does not include the preprocessing time, since we find that the preprocessing is usually at most 0.5 seconds.

We will discuss the computational results, which are divided into two parts. In the first part, we compare the five models `EF`, `F0`, `F`, `SF`, and `RF`. We evaluate the performance metrics of these models. The second part compares `RF` and `SFD`, quantifying the facilities that are saved by allowing continuous location. In the following, we will analyze the aggregated results, and we refer to Table 5 in the appendix for the detailed instance-wise results.

Benchmark	EF				F0				F				SF				RF			
	time	$\sigma\%$	$v_r\%$	S/A/T	time	$\sigma\%$	$v_r\%$	S/A/T	time	$\sigma\%$	$v_r\%$	S/A/T	time	$\sigma\%$	$v_r\%$	S/A/T	time	$\sigma\%$	$v_r\%$	S/A/T
City	1800.0	100.0	100.0	0/0/9	1801.71	56.8	83.3	0/3/9	1802.85	29.52	62.24	0/9/9	1801.31	30.1	66.91	0/9/9	1804.37	16.17	54.13	0/9/9
Kgroup_A	1800.0	100.0	100.0	0/0/11	1802.59	25.05	85.01	0/11/11	1803.0	33.14	82.17	0/11/11	1801.31	32.03	80.59	0/11/11	1622.56	21.48	77.52	1/11/11
Kgroup_B	1800.0	100.0	100.0	0/0/12	1800.37	92.55	98.79	0/1/12	1800.64	80.82	240.45	0/12/12	1801.44	79.72	191.88	0/12/12	1800.87	59.09	154.19	0/12/12
Random_A	1800.0	100.0	100.0	0/0/12	16.82	15.91	54.76	9/12/12	20.2	16.49	54.33	9/12/12	15.74	16.34	54.69	9/12/12	15.89	8.1	54.33	9/12/12
Random_B	1800.0	100.0	100.0	0/0/12	1317.65	36.38	63.29	1/12/12	1574.24	38.78	64.87	1/12/12	1501.22	39.95	67.46	1/12/12	1304.33	38.48	63.78	1/12/12
All	1800.0	100.0	100.0	0/0/56	625.8	37.35	74.78	10/39/56	675.0	35.21	86.18	10/56/56	637.57	35.46	83.59	10/56/56	604.9	23.7	75.42	11/56/56

Table 3: Results for continuous models

7.2. Comparative Analysis of Continuous Models

We compare five continuous models for the CSCP_δ, namely EF, F0, F, SF, and RF. For each benchmark and model, we record a triple of integers S/A/T: S denotes the number of solved instances, A denotes the number of affected instances, and T denotes the number of total instances in the benchmark. Moreover, we also report the average SGMs of the dual gaps, solving times and relative primal bounds among all instances in the benchmark. Table 3 summarizes these results.

First, we notice that EF cannot affect any instance in any benchmark; RF, SF and RF can affect all instances, i.e., solutions are provided by these models; RF is the model that solves the most number of instances (11), and SF is the second best one (10).

Secondly, we compare EF and F0. F0 is obviously superior to EF. With F0, 39 among 56 instances can be read by the CPLEX solver, while the instances modeled by EF are too large to read. Therefore, better solutions than trivial solutions are found by F0: on average, F0 finds solutions that use 24.3% fewer facilities than the trivial solution.

Then, we compare F0, F and SF. With the delimitation of complete and partial covering sets, F and SF can affect all instances (especially those in Kgroup_B, of which F0 could just read one). With the strengthening technique, SF has only marginal improvement in the relative primal bound, and solving time, while F is even slightly better than SF in the dual gap. We observe, in our experiment, that adding valid inequalities might slow down the internal solving process of CPLEX.

Finally, we compare SF and RF. RF outperforms SF in all performance metrics. Moreover, RF is the best one among those models affecting all instances. Indeed, for many instances, their degree-two-free networks may contain long edges, and RF avoids introducing too many variables and constraints for modeling their coverage.

Kgroup_B is the hardest benchmark, since the best model RF still has an average dual gap of 59.09%, and 154.19% relative primal bound. This means that RF cannot produce better solutions than the trivial one.

In Figure 7, we show scatter plots of the relative dual gaps and the relative primal bounds of affected instances between different settings. For every plot, there is a line in which the points have equal (X,Y)-values. If points fall below the line, then the Y-axis model performs better for the corresponding instances. Note that when comparing F0 and F, the plots do not consider the unaffected instances of F0 which are affected or solved by F. Moreover, F0 even closes more duality gaps than F, but F can find better primal solutions. These plots give an overview of all affected instances and support the above analysis.

To summarize, we have shown that the two proposed techniques— that to delimit the coverage areas from a given point in Section 2, and that to cover long edges in Section 6,— can reduce the model size drastically. Among the five models tested, RF features the best overall performance, which is achieved by directly modeling covers on long edges. On the other hand, delimiting the covering sets to the potential, complete and partial covers also reduces the model size, which allows F to read all the tested instances.

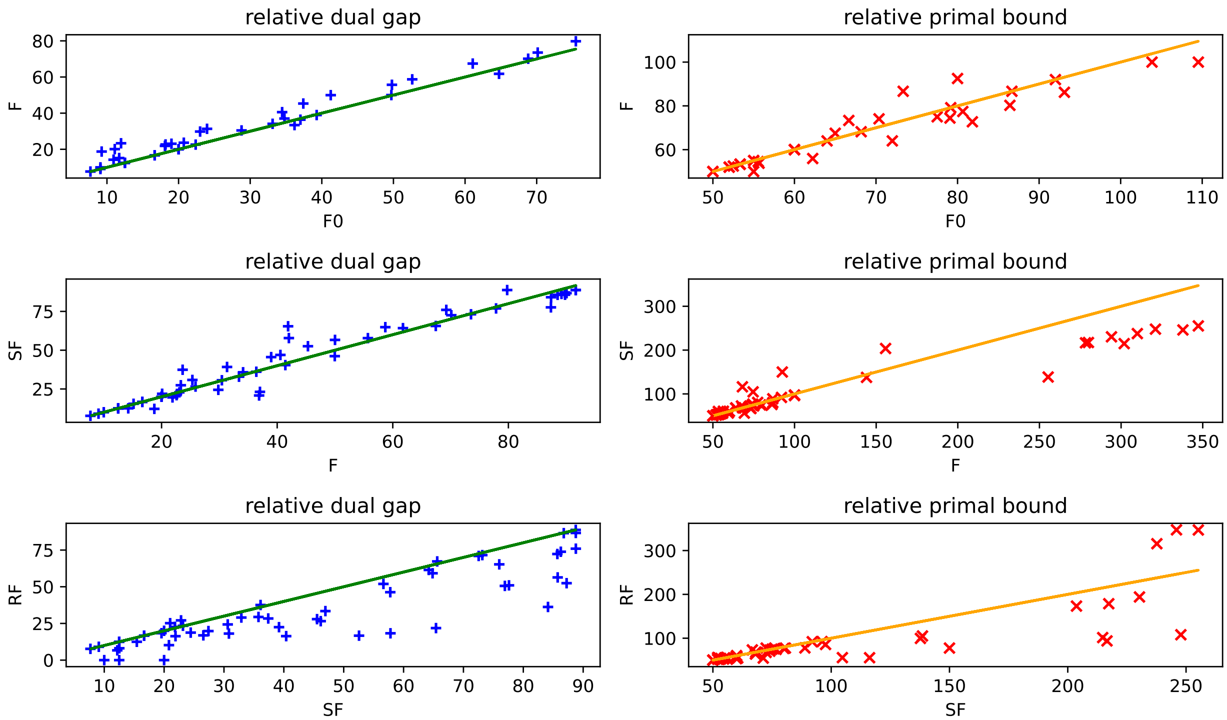


Figure 7: Scatter plots of the relative dual gaps and the relative primal bounds between different settings

7.3. Comparative Analysis of Continuous and Discrete Models

In CSCP$_\delta$, the facilities are located either at nodes or edges, while in the discrete variant considered in this section facilities can only be located at nodes. Our objective is to evaluate the number of facilities that can be saved by allowing continuous location. Since the discrete model studied here, `SFD`, is a discrete restriction of CSCP$_\delta$, every optimal solution is a feasible solution of CSCP$_\delta$. We solve `SFD` for the discrete facility covering problem and compare the results with the best model for CSCP$_\delta$, `RF`.

In addition to the previous performance statistics, we also record for each instance, a new relative primal bound for the continuous model defined as:

$$v'_r := \frac{\overline{v}}{\overline{v_d}} \times 100,$$

where $\overline{v_d}$ is the best solution found by `SFD`. If $v_r < 100\%$, then the continuous model (in this case, `RF`) finds a solution better than the one found by the discrete model.

Table 4 depicts some comparative results. A first observation is that `SFD` has fewer variables and constraints than `RF`, as it models a simpler problem. In addition, our strengthening techniques explain that `SFD` can solve almost all instances in a very short time. Moreover, even the average relative primal bound of `SFD` is smaller than `RF`. However, we note that, with the exception of `Kgroup_B`, `RF` finds solutions with (around 10%) fewer facilities. For `Kgroup_B`, `RF` has a large average dual gap, while `SFD` solves the instances to optimality.

In Figure 8, we also show scatter plots of the relative dual gaps (σ) and the relative primal bounds (v_r) for those instances affected by both `SFD` and `RF`. These plots complement the averaged results of Table 4 by giving information on all affected instances, and support the above analysis.

Benchmark	RF					SFD				
	time	$\sigma\%$	$v_r\%$	$v'_r\%$	S/A/T	time	$\sigma\%$	$v_r\%$	$v'_r\%$	S/A/T
`City`	1804.37	16.17	54.13	89.31	0/9/9	0.17	0.32	60.61	100.0	9/9/9
`Kgroup_A`	1622.56	21.48	77.52	91.11	1/11/11	0.53	2.71	85.08	100.0	11/11/11
`Kgroup_B`	1800.87	59.09	154.19	185.02	0/12/12	66.08	0.77	83.33	100.0	10/12/12
`Random_A`	15.89	8.1	54.33	85.96	9/12/12	0.02	1.16	63.21	100.0	12/12/12
`Random_B`	1304.33	38.48	63.78	91.82	1/12/12	0.97	2.11	69.45	100.0	12/12/12
`All`	604.9	23.7	75.42	104.61	11/56/56	2.18	1.3	72.07	100.0	54/56/56

Table 4: Results for continuous and discrete models

By allowing location at edges, the continuous model can reduce the number of installed facilities. However, it becomes more challenging to solve the problem. The results suggest that calling `SFD` and passing its solution as a warm-start to `RF` can make sense as a two-step optimization approach.

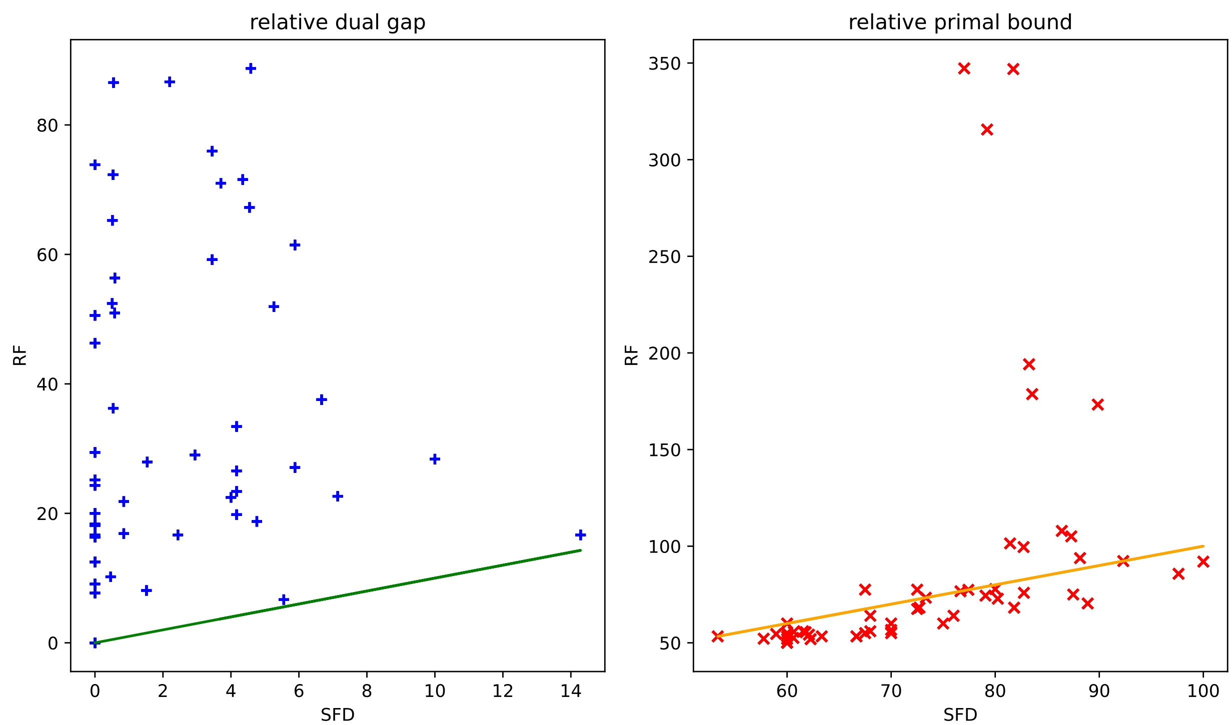


Figure 8: Scatter plots of the relative dual gaps and the relative primal bounds between **SFD** and **RF**

8. Conclusions

In this work, we use an integer programming approach to solve CSCP_δ, propose various MILP formulations for this problem and test these formulations against an existing MILP formulation on several benchmarks from the literature.

Specifically, we devise and implement four models for CSCP_δ: `F0`, `F` and `SF`, which belong to the same family of models, and `RF`, which is the reduced one. All of these models outperform the existing MILP model from [16], which is too large to be read into the MILP solver. We find the MILP size is the main barrier to scalability. The delimitation of those parts of the network that can be covered from a specific location has been revealed as a very effective technique to reduce the model size. In addition, avoiding breaking long edges in the reduced model also results in better scalability. In conclusion, `RF` is the best model: it can find good solutions with a small dual gap.

Meanwhile, the model `SF` is easily casted into `SFD` for the discrete restriction of CSCP_δ. We find that allowing continuous facilities decreases the number of installed facilities but increases the solving time significantly. We note that `SFD` finds an optimal solution for the discrete facility covering problem quickly, which is a primal solution for CSCP_δ. Therefore, `SFD` can be called as a fast MILP-based primal heuristic for CSCP_δ.

As for future studies, devising efficient heuristics to be integrated into MILP solvers can be useful to improve the primal performance of the proposed models. For instance, different relaxations of CSCP_δ can be worth exploring, such as that where demand only happens at nodes (i.e. only nodes are to be covered). Every solution of CSCP_δ would be a solution of such relaxation, and hence the optimal value of the latter is a valid dual lower bound of the optimal value of CSCP_δ. If solving this combinatorial relaxation is efficient and provides a stronger dual lower bound than the LP relaxation of CSCP_δ, we can utilize this result and integrate the combinatorial dual bound into the MILP solver, which leads to a combinatorial branch-and-bound algorithm.

Acknowledgments

The authors would like to thank Joerg Kalcsics for providing access to some of the real-world network instances used for the experiments.

Funding source

The research of Liding Xu is supported by French Ministry of Higher Education and Research (MESRI). The research of Mercedes Pelegrín was supported by the Chair "Integrated Urban Mobility", backed by L'X - École Polytechnique and La Fondation de l'École Polytechnique. The Partners of the Chair shall not under any circumstances accept any liability for the content of this publication, for which the author shall be solely liable.

References

[1] B. Adenso-Díaz, F. Rodríguez, A simple search heuristic for the mclp: application to the location of ambulance bases in a rural region, Omega 25 (1997) 181–187.

[2] V. Gusev, The vertex cover game: Application to transport networks, Omega 97, 102102.

[3] S. Shirinivas, S. Vetrivel, N. Elango, Applications of graph theory in computer science an overview, International Journal of Engineering Science and Technology 2 (9) (2010) 4610–4621.

[4] D. Briskorn, M. Dienstknecht, Covering polygons with discs: The problem of crane selection and location on construction sites, Omega 97, 102114.

[5] P. R. Jenkins, B. J. Lunday, M. J. Robbins, Robust, multi-objective optimization for the military medical evacuation location-allocation problem, Omega 97 (2020) 102088.

[6] J. Bell, S. Griffis, W. Cunningham, J. Eberlan, Location optimization of strategic alert sites for homeland defense, Omega 39 (2011) 151–158.

[7] L. Xu, Optimal location of safety landing sites, Tech. rep., LIX, École Polytechnique, hal-03286640 (2021).

[8] R. Church, C. Revelle, The maximal covering location problem, Papers of the Regional Science Association 32 (1979) 101–118.

[9] C. Toregas, C. ReVelle, Optimal location under time or distance constraints, Papers of the Regional Science Association 28 (1) (1972) 131–143.

[10] F. Plastria, Continuous covering location problems, in: Z.Drezner, H.Hamacher (Eds.), Facility location: applications and theory, Springer, 2002, pp. 37–79.

[11] S. García, A. Marín, Covering location problems, in: G. Laporte, S. Nickel, F. S. da Gama (Eds.), Location Science, Springer, 2nd edition, 2019, pp. 99–119.

[12] R. Blanquero, E. Carrizosa, G. Boglárka, Maximal covering location problems on networks with regional demand, Omega 64 (2016) 77–85.

[13] O. Berman, J. Kalcsics, D. Krass, On covering location problems on networks with edge demand, Computers & Operations Research 74 (2016) 214–227.

[14] M. Baldomero-Naranjo, J. Kalcsics, A. Rodríguez-Chía, Minmax regret maximal covering location problems with edge demands, Computers & Operations Research 130 (2021) 105181.

[15] Y. Gurevich, L. Stockmeyer, U. Vishkin, Solving NP-hard problems on graphs that are almost trees and an application to facility location problems, Journal of the ACM (JACM) 31 (3) (1984) 459–473.

[16] N. Fröhlich, A. Maier, H. Hamacher, Covering edges in networks, Networks 75 (3) (2020) 278–290.

[17] T. Hartmann, S. Lendl, G. Woeginger, Continuous facility location on graphs, Mathematical Programming (2021) 1–21.

[18] R. Church, M. Meadows, Location modeling utilizing maximum service distance criteria, Geographical Analysis 11 (4) (1979) 358–373.

[19] T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein, Introduction to algorithms, MIT press, 2009.

[20] I. Ljubić, R. Weiskircher, U. Pferschy, G. W. Klau, P. Mutzel, M. Fischetti, An Algorithmic Framework for the Exact Solution of the Prize-Collecting Steiner Tree Problem, Mathematical Programming 105 (2) (2006) 427–449.

[21] A. A. Hagberg, D. A. Schult, P. J. Swart, Exploring network structure, dynamics, and function using networkx, in: G. Varoquaux, T. Vaught, J. Millman (Eds.), Proceedings of the 7th Python in Science Conference, Pasadena, CA USA, 2008, pp. 11 – 15.

[22] I. Dunning, J. Huchette, M. Lubin, Jump: A modeling language for mathematical optimization, SIAM Review 59 (2) (2017) 295–320.

Appendix

Table 5: Detailed experimental results for comparing different models.

time : the solving time
gap : the relative dual gap σ
primal : the primal bound $\underline{v}$

	Subdivided network		**Degree-two-free network**		EF			F0			F			SF			RF			SFD		
instance	nodes	edges	nodes	edges	time	gap	primal	time	gap	primal	time	gap	primal	time	gap	primal	time	gap	primal	time	gap	primal
city_479	584	689	428	533	NAN	NAN	NAN	NAN	NAN	NAN	1811.1	36.87	259.0	1801.5	20.81	210.0	1801.0	10.2	52.1	0.2	0.46	57.8
city_628	756	919	557	720	NAN	NAN	NAN	NAN	NAN	NAN	1800.1	42.03	347.0	1801.7	57.81	487.0	1803.5	18.29	55.7	0.4	0.0	60.6
city_268	307	380	229	302	NAN	NAN	NAN	NAN	NAN	NAN	1810.9	25.33	110.0	1800.6	30.83	118.0	1806.4	18.09	54.9	0.1	0.0	60.0
city_276	318	403	232	317	NAN	NAN	NAN	NAN	NAN	NAN	1800.1	41.89	130.0	1800.8	65.42	222.0	1805.9	21.83	55.5	0.2	0.85	61.8
city_265	298	373	221	296	NAN	NAN	NAN	NAN	NAN	NAN	1800.4	25.88	113.0	1802.7	26.56	107.0	1801.0	16.87	54.2	0.1	0.85	62.1
city_213	249	319	196	266	NAN	NAN	NAN	1811.4	28.79	89.0	1800.7	30.43	80.0	1800.3	30.63	80.0	1807.9	24.32	55.9	0.2	0.0	61.5
city_138	162	201	135	174	NAN	NAN	NAN	1801.4	18.1	55.0	1801.1	21.89	54.0	1800.6	19.56	54.0	1804.7	18.35	52.5	0.0	0.0	60.6
city_771	944	1123	755	934	NAN	NAN	NAN	NAN	NAN	NAN	1800.1	41.42	428.0	1801.1	40.39	422.0	1803.1	16.34	54.6	0.3	0.0	59.0
city_132	152	179	112	139	NAN	NAN	NAN	1802.7	10.92	59.0	1801.2	14.24	57.0	1802.5	12.52	56.0	1805.8	8.09	51.9	0.0	1.52	62.3
K100.3.con	94	191	66	163	NAN	NAN	NAN	1800.7	41.25	27.0	1805.3	49.96	26.0	1801.2	46.16	25.0	1800.5	26.56	92.3	3.4	4.17	92.3
K100.10	67	118	51	102	NAN	NAN	NAN	1800.8	11.97	19.0	1800.4	23.33	20.0	1801.4	27.41	20.0	1800.9	19.8	70.4	0.1	4.17	88.9
K100.6	50	92	38	80	NAN	NAN	NAN	1803.6	11.09	15.0	1800.9	20.1	15.0	1804.9	21.9	15.0	1807.3	16.31	68.2	0.1	0.0	72.7
K100.con	145	291	121	267	NAN	NAN	NAN	1800.2	49.79	39.0	1806.8	55.7	39.0	1800.2	57.78	40.0	1800.1	46.29	77.8	0.6	0.0	80.0
K100.1	120	263	70	213	NAN	NAN	NAN	1800.2	37.41	46.0	1806.6	45.32	42.0	1800.1	52.56	41.0	1810.0	16.67	85.7	1.9	2.44	97.6
K100.5.con	86	175	64	153	NAN	NAN	NAN	1811.1	34.46	25.0	1800.4	40.56	24.0	1802.6	46.94	25.0	1802.9	33.41	77.4	0.4	4.17	77.4
K100.8.con	107	208	78	179	NAN	NAN	NAN	1800.6	36.19	34.0	1800.8	33.4	32.0	1800.5	32.9	33.0	1800.7	29.02	74.4	0.9	2.94	79.1
K100.9	56	104	35	83	NAN	NAN	NAN	1803.0	9.26	18.0	1805.8	18.75	16.0	1800.4	12.18	15.0	565.2	6.67	68.2	0.1	5.56	81.8
K100.4.con	85	169	60	144	NAN	NAN	NAN	1806.5	34.78	27.0	1801.7	37.02	25.0	1800.7	23.15	22.0	1804.5	23.39	75.9	0.2	4.17	82.8
K100.2	61	120	42	101	NAN	NAN	NAN	1800.9	23.01	19.0	1800.6	29.82	19.0	1802.2	24.44	18.0	1800.7	18.75	75.0	0.1	4.76	87.5
K100.7	74	142	48	116	NAN	NAN	NAN	1800.9	23.97	23.0	1803.7	31.33	23.0	1800.4	39.19	23.0	1801.7	22.46	92.0	0.1	4.0	100.0
K400.1.con	587	1224	379	1016	NAN	NAN	NAN	NAN	NAN	NAN	1800.5	69.26	338.0	1800.7	75.99	442.0	1800.5	65.26	173.3	15.6	0.51	89.9
K400.2	709	1429	522	1242	NAN	NAN	NAN	NAN	NAN	NAN	1800.6	89.77	732.0	1800.2	87.23	565.0	1801.3	52.43	107.9	32.4	0.51	86.4
K400.10.con	671	1373	486	1188	NAN	NAN	NAN	NAN	NAN	NAN	1801.7	89.19	650.0	1800.7	86.27	509.0	1800.1	73.85	194.1	27.4	0.0	83.3
K400.7.con	633	1275	462	1104	NAN	NAN	NAN	NAN	NAN	NAN	1801.2	88.51	630.0	1800.2	85.69	489.0	1802.2	72.31	178.7	140.4	0.53	83.6
K200.con	184	374	121	311	NAN	NAN	NAN	1804.4	39.28	70.0	1800.4	38.96	65.0	1805.2	45.55	60.0	1800.7	27.93	72.8	0.6	1.54	80.2
K400.5.con	600	1179	431	1010	NAN	NAN	NAN	NAN	NAN	NAN	1800.3	77.87	317.0	1800.2	76.92	303.0	1801.4	50.58	99.5	43.6	0.0	82.7
K400.8.con	749	1501	610	1362	NAN	NAN	NAN	NAN	NAN	NAN	1800.0	91.64	794.0	1800.0	88.76	578.0	1801.2	88.74	347.2	1800.3	4.58	77.0
K400.4	516	1103	358	945	NAN	NAN	NAN	NAN	NAN	NAN	1800.1	87.36	503.0	1802.3	77.61	273.0	1800.7	50.95	105.1	17.3	0.58	87.3
K400.9.con	588	1239	363	1014	NAN	NAN	NAN	NAN	NAN	NAN	1800.1	87.38	587.0	1803.5	84.13	457.0	1801.5	36.23	93.8	23.8	0.54	88.2
K400	715	1398	568	1251	NAN	NAN	NAN	NAN	NAN	NAN	1801.2	90.06	716.0	1800.2	86.76	549.0	1800.1	86.53	315.6	371.9	0.55	79.2
K400.6.con	789	1583	612	1406	NAN	NAN	NAN	NAN	NAN	NAN	1800.0	91.67	837.0	1803.0	88.77	615.0	1800.6	86.67	346.9	1801.0	2.2	81.7
K400.3.con	595	1191	450	1046	NAN	NAN	NAN	NAN	NAN	NAN	1801.4	89.8	634.0	1801.0	85.74	451.0	1800.1	56.35	101.4	96.1	0.58	81.4
r_20_0.3_49	46	75	46	75	NAN	NAN	NAN	1801.4	18.18	11.0	1801.6	22.67	11.0	1800.6	21.82	11.0	1800.7	22.64	55.0	0.0	7.14	70.0
r_15_0.1_14	19	18	15	14	NAN	NAN	NAN	0.0	12.5	8.0	0.0	12.5	8.0	0.0	12.5	8.0	0.0	0.0	53.3	0.0	0.0	53.3
r_15_0.3_25	26	36	21	31	NAN	NAN	NAN	20.2	12.5	8.0	56.8	12.5	8.0	23.9	12.5	8.0	13.6	12.5	53.3	0.0	0.0	60.0
r_10_0.1_12	16	18	14	16	NAN	NAN	NAN	0.1	20.0	5.0	0.1	20.0	5.0	0.0	20.0	5.0	0.0	20.0	50.0	0.0	0.0	60.0
r_20_0.2_34	39	53	33	47	NAN	NAN	NAN	336.8	9.09	11.0	376.5	9.09	11.0	159.1	9.09	11.0	178.8	9.09	55.0	0.0	0.0	60.0
r_10_0.2_9	12	11	5	4	NAN	NAN	NAN	0.0	20.0	5.0	0.0	20.0	5.0	0.0	20.0	5.0	0.0	0.0	50.0	0.0	0.0	60.0
r_10_0.3_12	15	17	13	15	NAN	NAN	NAN	0.3	16.67	6.0	0.2	16.67	6.0	0.1	16.67	6.0	0.2	16.67	60.0	0.0	14.29	70.0

Table 5 continued

instance	Subdivided network nodes	Subdivided network edges	Degree-two-free network nodes	Degree-two-free network edges	EF time	EF gap	EF primal	F0 time	F0 gap	F0 primal	F time	F gap	F primal	SF time	SF gap	SF primal	RF time	RF gap	RF primal	SFD time	SFD gap	SFD primal
r_10_0.4_13	14	17	14	17	NAN	NAN	NAN	0.2	16.67	6.0	1.2	16.67	6.0	0.4	16.67	6.0	1.0	16.67	60.0	0.0	0.0	60.0
r_15_0.4_45	40	70	40	70	NAN	NAN	NAN	1800.7	20.73	8.0	1808.2	23.64	8.0	1801.2	37.4	9.0	1800.9	28.37	53.3	0.1	10.0	66.7
r_20_0.1_23	29	32	23	26	NAN	NAN	NAN	2.1	9.09	11.0	2.8	10.0	10.0	1.0	10.0	10.0	2.4	0.0	50.0	0.0	0.0	60.0
r_20_0.4_69	52	101	52	101	NAN	NAN	NAN	1801.3	37.01	12.0	1801.0	36.39	12.0	1803.3	36.11	12.0	1801.1	37.56	60.0	0.1	6.67	75.0
r_15_0.2_22	25	32	21	28	NAN	NAN	NAN	3.4	12.5	8.0	4.6	12.5	8.0	7.0	12.5	8.0	3.2	12.5	53.3	0.0	0.0	66.7
r_40_0.3_219	149	328	149	328	NAN	NAN	NAN	1809.4	68.81	31.0	1800.7	70.13	30.0	1802.9	72.54	29.0	1806.4	70.99	77.5	3.1	3.7	67.5
r_30_0.1_54	52	76	46	70	NAN	NAN	NAN	1801.6	11.71	16.0	1801.9	15.15	16.0	1802.4	15.44	16.0	1802.4	12.5	53.3	0.0	0.0	63.3
r_40_0.1_84	78	122	76	120	NAN	NAN	NAN	1801.4	22.39	21.0	1801.5	22.6	21.0	1800.7	21.01	21.0	1801.5	25.18	55.0	0.1	0.0	67.5
r_30_0.2_75	67	112	67	112	NAN	NAN	NAN	1800.7	33.13	18.0	1801.0	34.1	18.0	1800.4	35.77	18.0	1801.6	29.41	56.7	0.2	0.0	70.0
r_30_0.4_188	117	275	117	275	NAN	NAN	NAN	1800.4	70.14	22.0	1800.2	73.53	26.0	1801.7	73.16	24.0	1800.7	71.56	76.7	5.4	4.35	76.7
r_40_0.4_297	191	448	191	448	NAN	NAN	NAN	1800.5	75.47	32.0	1805.8	79.79	37.0	1800.1	88.78	60.0	1800.2	75.95	77.5	22.2	3.45	72.5
r_25_0.4_112	83	170	83	170	NAN	NAN	NAN	1800.1	64.74	18.0	1806.2	61.76	16.0	1801.0	64.22	17.0	1800.2	61.45	64.0	0.3	5.88	68.0
r_25_0.2_58	53	86	53	86	NAN	NAN	NAN	1801.2	19.01	13.0	1800.8	23.08	13.0	1801.9	22.83	13.0	1800.8	27.08	56.0	0.1	5.88	68.0
r_25_0.3_98	74	147	74	147	NAN	NAN	NAN	1800.7	49.71	16.0	1811.2	50.01	16.0	1800.4	56.63	17.0	1804.9	51.93	64.0	0.2	5.26	76.0
r_30_0.3_131	92	193	92	193	NAN	NAN	NAN	1800.8	61.07	20.0	1800.1	67.44	22.0	1800.3	65.62	20.0	1800.5	67.26	73.3	0.7	4.55	73.3
r_40_0.2_148	112	220	112	220	NAN	NAN	NAN	1800.3	52.62	26.0	1800.6	58.71	27.0	1800.2	64.84	29.0	1802.5	59.19	67.5	0.5	3.45	72.5
r_25_0.1_36	40	51	36	47	NAN	NAN	NAN	41.3	7.69	13.0	354.0	7.69	13.0	201.9	7.69	13.0	36.4	7.69	52.0	0.0	0.0	60.0