http://pubsonline.informs.org/journal/ijoc

INFORMS JOURNAL ON COMPUTING
Vol. 32, No. 4, Fall 2020, pp. 952–967
ISSN 1091-9856 (print), ISSN 1526-5528 (online)

The Optimal Design of Low-Latency Virtual Backbones

Hamidreza Validi,[a] Austin Buchanan[a]

[a] School of Industrial Engineering and Management, Oklahoma State University, Stillwater, Oklahoma 74078

Contact: hamidreza.validi@okstate.edu, http://orcid.org/0000-0002-7983-7262 (HV); buchanan@okstate.edu, http://orcid.org/0000-0003-2999-9666 (AB)

Received: July 7, 2017
Revised: May 24, 2018; February 15, 2019; May 24, 2019
Accepted: June 3, 2019
Published Online in Articles in Advance: April 15, 2020

https://doi.org/10.1287/ijoc.2019.0914

Abstract. Two nodes of a wireless network may not be able to communicate with each other directly, perhaps because of obstacles or insufficient signal strength. This necessitates the use of intermediate nodes to relay information. Often, one designates a (preferably small) subset of them to relay these messages (i.e., to serve as a virtual backbone for the wireless network), which can be seen as a connected dominating set (CDS) of the associated graph. Ideally, these communication paths should be short, leading to the notion of a latency-constrained CDS. In this paper, we point out several shortcomings of a previously studied formalization of a latency-constrained CDS and propose an alternative one. We introduce an integer programming formulation for the problem that has a variable for each node and imposes the latency constraints via an exponential number of cut-like inequalities. Two nice properties of this formulation are that (1) it applies when distances are hop-based and when they are weighted and (2) it easily generalizes to ensure fault tolerance. We provide a branch-and-cut implementation of this formulation and compare it with a new polynomial-size formulation. Computational experiments demonstrate the superiority of the cut-like formulation. We also study related questions from computational complexity, such as approximation hardness, and answer an open problem regarding the fault diameter of graphs.

History: Accepted by David Alderson, Area Editor for Network Optimization: Algorithms and Applications; S. Raghavan, former Area Editor.
Funding: This work was supported by the Division of Civil, Mechanical and Manufacturing Innovation of the National Science Foundation [Grant 1662757].
Supplemental Material: The online supplement is available at https://doi.org/10.1287/ijoc.2019.0914.

Keywords: connected dominating set • strongly connected dominating set • latency • delay • hop constraint • k-club • length-bounded cut • wireless networks • fault-tolerant • integer programming

1. Introduction

Two nodes of a wireless network may not be able to communicate with each other *directly* perhaps because of obstacles or insufficient signal strength. This necessitates the use of intermediate nodes to relay information. Often, one designates a small subset of them to relay messages (i.e., to serve as a virtual backbone for the wireless network), which amounts to a connected dominating set of the associated graph, defined below.

Definition 1 (CDS). A subset $D \subseteq V$ of vertices is a connected dominating set (CDS) for an undirected graph $G = (V, E)$ if

1. D is *dominating*, that is, every vertex from $V \setminus D$ neighbors a vertex of D; and
2. D is *connected*, that is, the subgraph $G[D]$ induced by D is connected.

If the graph G is not complete,[1] a CDS can be equivalently defined as a subset $D \subseteq V$ of vertices such that, for every vertex pair $\{a, b\} \in \binom{V}{2}$, there exists a path connecting a and b whose interior vertices belong to D.

A CDS ensures that the nodes of the network can communicate with each other. It provides little guarantee on *how long* it will take for a message to be received once it has been sent. This has led some researchers to impose additional constraints on the CDS $D \subseteq V$, namely, that the subgraph induced by the dominating set D has a diameter of at most s; that is, it is a dominating s-club (Li et al. 2008, Zhang et al. 2008, Buchanan et al. 2014). This ensures that messages will be received in $s + 2$ hops: one hop to reach the CDS, at most s hops within the CDS, and one hop to reach the destination.

Definition 2 (Dominating s-Club). A subset $D \subseteq V$ of vertices is a dominating s-club for an undirected graph $G = (V, E)$ if

1. D is *dominating*; and
2. D is an *s-club*; that is, the subgraph $G[D]$ induced by D has a diameter of at most s.

We argue that this formalization of the problem is less than ideal. First, and most importantly, a dominating s-club does not quite capture the intent of the hop

constraints, as we will illustrate. Suppose that we want a CDS that facilitates 4-hop communication in the graph in Figure 1. This can be ensured by the dominating 2-club given in Figure 1(a). Indeed, a message sent from node 5 to node 8 through this virtual backbone must follow the path 5-4-3-6-8, which takes four hops.

If 3-hop communication were required, one might search for a dominating 1-club, but none exist in this graph. This may lead us to believe that a CDS that facilitates 3-hop communication does not exist, but this belief would be false. Indeed, in Figure 1(b) we provide a CDS which needs at most *two* hops to transmit information, so we could call it a latency-2 CDS. Further, Figure 1(c) gives a latency-3 CDS, which is also a minimum CDS! Note that a message can be *directly* passed from node 8 to node 9 in the wireless network because they are adjacent; it does *not* have to be relayed through the CDS nodes.

Another limitation of previous works is that they make the simplifying assumption that distances are measured by the number of hops (see Li et al. 2008; Zhang et al. 2008; Du and Wan 2013, chapter 7; and Buchanan et al. 2014). However, this may ultimately provide a poor approximation to the actual end-to-end delay when the delays at the nodes differ. For example, a particular node may play a central role in the CDS, needing to relay a large number of messages. This may cause messages to have to wait to be transmitted, and these *queueing delays* may be more realistically captured for our purposes via *node-weighted* delays as opposed to hop-based delays. For more information about this and other delays in wireless networks, consult Xie and Haenggi (2009) and Zhong et al. (2017).

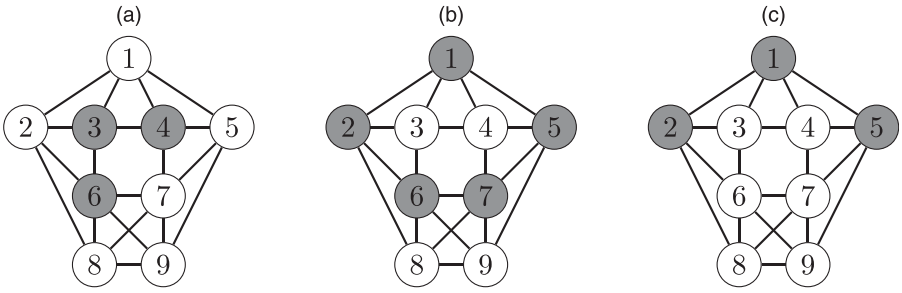
With these shortcomings in mind, we propose a new formalization of a low-latency virtual backbone, which we call a latency-s CDS. For purposes of generality, it is defined in terms of a *directed* and—without loss of generality—*edge-weighted* graph. By allowing for directed edges, we can model nonuniform transmission ranges. For example, consider the case in which a node i has a large transmission range and is far away from a node j that has a small transmission range. In this scenario, the edge (i,j) should exist, but not the edge (j,i). Note that, as we make the transition to directed graphs, we are no longer referring to a "CDS" in the sense of Definition 1, but rather in terms of a *strongly connected dominating set* in the sense of Li et al. (2009). This is defined as a subset D of vertices such that (a) D induces a strongly connected subgraph and (b) every vertex from $V \setminus D$ has both an in-neighbor and an out-neighbor in D.

Definition 3 (Latency-s CDS). A vertex subset $D \subseteq V$ is a latency-s CDS for a directed graph $G = (V, E)$ under edge weights $w : E \to \mathbb{R}_+$ if, for every vertex pair $(a, b) \in V \times V$, there is a path from a to b of length at most s whose interior vertices belong to D.

Observe that the graph G in Definition 3 is edge weighted, but not vertex weighted. This is without loss of generality, as the following will illustrate. Consider the 3-vertex, undirected path graph 1-2-3 representing a wireless network. Sending a message from node 1 to node 3 would incur delays at nodes 1 and 2 (because the delay is based on the transmitting node), as well as delays on edges $\{1,2\}$ and $\{2,3\}$, for an end-to-end delay that might be denoted $d_1 + d_2 + d_{\{1,2\}} + d_{\{2,3\}}$. Instead, we can replace each undirected edge $\{i,j\}$ by its directed counterparts (i,j) and (j,i) and let the delay of each directed edge $d_{(i,j)}$ be the delay of its undirected counterpart $d_{\{i,j\}}$ plus the delay of its tail node d_i. In this way, it is sufficient to consider a directed graph with only edge weights.

Definition 3 overcomes the aforementioned issues with the previous formalization based on dominating s-clubs. As an added bonus, it has superior computational properties. Indeed, checking whether a graph admits a latency-s CDS is as simple as checking whether the graph's diameter is at most s. In contrast, the problem of checking whether there exists *any* dominating s-club is NP-complete; specifically, this is true under hop-based distances for the two most restrictive (but nontrivial) cases in which $s = \text{diam}(G) - 2$ (Schaudt 2013) and $s = \text{diam}(G) - 1$ (Buchanan et al. 2014), where diam denotes the graph's diameter.

Figure 1. Example Low-Latency Virtual Backbones



Notes. (a) Dominating 2-club. (b) Latency-2 CDS. (c) Latency-3 CDS.

The associated optimization problem is as follows.

Problem: The minimum latency-s CDS problem.

Input: A directed graph $G = (V, E)$, a weight $w_e \geq 0$ for each edge $e \in E$, and a number s.

Output: (if any exist) A smallest subset $D \subseteq V$ of vertices that is a latency-s CDS.

In this paper, we propose an integer programming (IP) formulation for this problem that uses an exponential number of cut-like inequalities. As we will see, a relatively simple implementation of it significantly outperforms a polynomial-size formulation that we introduce. This second formulation has $O(sn^2)$ variables and $O(snm)$ constraints and applies when the distances are hop-based, where n and m denote the number of vertices and edges, respectively. In contrast, the cut-like formulation applies when there are weighted delays.

1.1. Previous Work

The minimum CDS problem is a well-studied NP-hard problem (Garey and Johnson 1979) in which the task is to find a CDS of minimum cardinality. For example, Figure 2(a) provides a CDS and Figure 2(b) provides a *minimum* CDS. The reader is encouraged to consult the book by Du and Wan (2013) for motivating applications, approximation algorithms, and hardness results. There are a number of IP formulations and implementations for the minimum CDS problem and for the equivalent maximum-leaf spanning tree problem (Fujie 2004, Morgan and Grout 2008, Lucena et al. 2010, Simonetti et al. 2011, Fan and Watson 2012, Gendron et al. 2014, Buchanan et al. 2015). See also the literature on the regenerator location problem (Chen et al. 2010, 2015; Li and Aneja 2017). To our knowledge, the state-of-the-art[2] IP formulation and implementation are due to Fujie (2004) and Buchanan et al. (2015), respectively, although several of the previously mentioned approaches work well. As far as we know, the only previous work to propose an IP formulation for a latency-constrained variant of the CDS problem is by Buchanan et al. (2014); however, it is for dominating s-clubs.

Assuming the input graph is not complete, the minimum CDS problem can be formulated as an IP as follows, where x_i is a binary variable representing the decision to include vertex i in the CDS. A vertex cut is a subset $C \subset V$ of vertices such that $G - C := G[V \setminus C]$ is disconnected and nontrivial (i.e., has at least two nodes).

$$\min \sum_{i \in V} x_i \tag{1}$$

$$\sum_{i \in C} x_i \geq 1, \quad \forall \text{ vertex cut } C \subset V \tag{2}$$

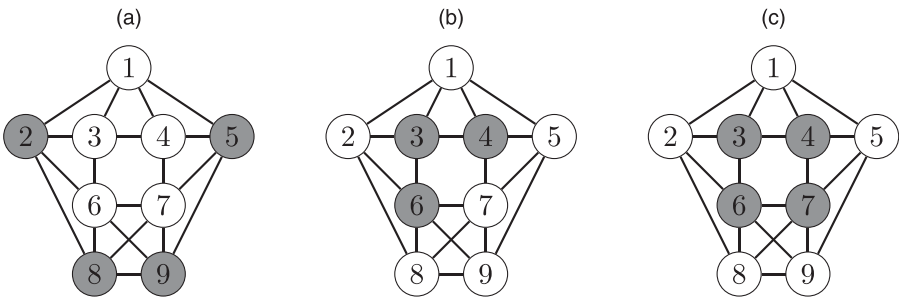
$$x_i \in \{0, 1\}, \quad \forall i \in V. \tag{3}$$

This particularly elegant formulation is essentially due to Fujie (2004), and its linear programming relaxation can be solved in polynomial time despite having exponentially many constraints, as the separation problem for the vertex cut constraints (2) can be solved in polynomial time. An implementation of this formulation by Buchanan et al. (2015) solved 42 of 47 standard test instances, each in under 10 seconds (and never taking longer than 500 seconds), whereas no earlier approach solved 42 instances each in a 1-hour time limit. In their implementation, Buchanan et al. (2015) add violated vertex cut inequalities on-the-fly, cutting off infeasible *integer* points. Our proposed formulation in this paper, which generalizes Fujie's formulation, is implemented in the same manner.

One drawback of a CDS is that it can be vulnerable to node or arc failures. For example, consider the minimum CDS from Figure 2(b). If node 3 fails, this renders the virtual backbone inoperative as it no longer can relay information (say, from node 5 to node 8).

This motivates the notion of a fault-tolerant CDS—one that remains a CDS when fewer than k nodes fail. This has been called a k-connected k-dominating set (k-k-CDS) as it can equivalently be defined as a subset $S \subseteq V$ of vertices such that $G[S]$ is k-vertex-connected and every vertex of $V \setminus S$ has k neighbors in S. Figure 2(c) gives a 2-2-CDS, which remains a CDS if one vertex fails. The associated optimization problem, the minimum

Figure 2. Example Virtual Backbones



Notes. (a) CDS. (b) *Minimum* CDS. (c) 2-connected 2-dominating set.

k-k-CDS problem, admits the following formulation (Ahn and Park 2015, Buchanan et al. 2015).

$$\min \sum_{i \in V} x_i \tag{4}$$

$$\sum_{i \in C} x_i \geq k, \quad \forall \text{ vertex cut } C \subset V \tag{5}$$

$$x_i \in \{0,1\}, \quad \forall i \in V. \tag{6}$$

The formulations that we propose in this paper generalize this k-k-CDS formulation and Fujie's CDS formulation.

1.2. Notation and Terminology

From now on, unless stated otherwise, $G = (V, E)$ will be a directed graph, with vertex set V and edge set $E \subset V \times V$, that has no loops and no parallel edges. By "no parallel edges," we mean that there is at most one directed edge from a vertex i to a vertex j, and so we can refer to it by the notation (i, j). Here, i is called the tail and j is the head. Frequently, we *bidirect* an undirected edge, which we define to be the operation in which an undirected edge $\{i, j\}$ is replaced by its directed counterparts (i, j) and (j, i). When the edges of G are reciprocated, that is, if $(i, j) \in E$ implies $(j, i) \in E$, then we say that G is bidirected—not to be confused with the bidirected graphs of Edmonds and Johnson (1970); see also Schrijver (2003).

For each edge $e \in E$ of G, there is an associated nonnegative weight w_e representing the delay. In the *hop-based* case, each weight is one. The distance from vertex a to vertex b in graph G, denoted $\text{dist}_G(a, b)$, is the length of a shortest path from a to b in G, edge-weighted by w. Convention states that if there is no path from a to b in G, then $\text{dist}_G(a, b) = \infty$. The diameter of G, denoted $\text{diam}(G)$, is the maximum of these pairwise distances, that is, $\text{diam}(G) := \max\{\text{dist}_G(a, b) \mid a, b \in V\}$.

The out-neighborhood and in-neighborhood of a vertex $v \in V$ in G are denoted $N_G^+(v) := \{w \in V \mid (v, w) \in E\}$ and $N^-(v) := \{u \in V \mid (u, v) \in E\}$, respectively. For a vertex subset S, $\delta_G^+(S)$ denotes the subset of edges whose tail belongs to S and whose head does not. Similarly, $\delta_G^-(S)$ denotes the subset of edges whose head belongs to S but whose tail does not. For a singleton $S = \{v\}$, let $\delta_G^+(v) := \delta_G^+(\{v\})$ and $\delta_G^-(v) := \delta_G^-(\{v\})$. When the graph G in question is clear, the subscripts G in $N_G^+(\cdot)$, $N_G^-(\cdot)$, $\delta_G^+(\cdot)$, and $\delta_G^-(\cdot)$ are omitted. The subset of edges having both endpoints in $S \subseteq V$ is denoted $E(S) := \{(i, j) \in E \mid i, j \in S\}$.

1.3. Our Contributions

In Section 2, we examine the complexity of latency-s CDS's. Specifically, we answer questions like: How quickly can one verify that a given subset of vertices is a latency-s CDS? And, how hard is the minimum latency-s CDS problem?

In Section 3, we propose IP formulations for the minimum latency-s CDS problem. The first formulation, which we call CUT, has n binary variables and an exponential number of cut-like constraints. We then generalize this formulation so that it models the fault-tolerant variant in which one seeks a latency-s CDS that maintains feasibility after a small number of vertex failures. Then we give a second IP formulation, which we call POLY, that has $O(sn^2)$ variables and $O(snm)$ constraints. It serves as a baseline for computational comparisons.

In Section 4, we examine the complexity of the separation problem associated with formulation CUT. Specifically, we show that, under hop-based distances, it is polynomial-time solvable for $s \in \{2, 3, 4\}$ and NP-hard when $s \geq 5$. En route to proving this, we answer an open question of Xu et al. (2005), by showing that it is indeed NP-hard to compute a graph's fault diameter.

In Section 5, we perform computational experiments. Our results demonstrate that a branch-and-cut implementation of formulation CUT significantly outperforms the polynomial-size formulation POLY. Notably, CUT makes easy work of a real-life instance with 300 nodes, whereas formulation POLY struggles to solve instances with 50 nodes in an hour.

In Section 6, we conclude and discuss directions for future research.

2. The Complexity of Latency-s CDS

In this section, we examine the complexity of latency-s CDS's. First, we pinpoint the complexity of verifying feasible solutions, showing essentially that a quadratic running time is unavoidable under a plausible complexity assumption. Then we establish the inapproximability of the minimum latency-s CDS problem.

2.1. The Complexity of Verifying Feasible Solutions

To verify that a given subset $D \subseteq V$ of vertices is a latency-s CDS, we can compute, for each vertex $v \in V$, the shortest paths from v to all other nodes $t \in V \setminus \{v\}$ and check that these paths are short enough. However, we are not interested in just any paths from v to t; these paths must not cross vertices from $V \setminus D$. In our proposed approach, we solve an instance of the single source shortest path problem (SSSP) in the subgraph $\overrightarrow{G}_v^D = (V, \overrightarrow{E}_v^D)$, which has edge set

$$\overrightarrow{E}_v^D := E(D) \cup \delta^+(D) \cup (\delta^+(v) \cap \delta^-(D)). \tag{7}$$

Here, we preserve the edges $E(D)$ that have both endpoints in D, those edges $\delta^+(D)$ that point out of D, and those edges $\delta^+(v) \cap \delta^-(D)$ whose tail is v and whose head is in D. This set $\overrightarrow{E}_v^D$ includes all edges that might be used in a suitable path from v to another node. Of course, we need not create the graph $\overrightarrow{G}_v^D$ in

the implementation, as nearly any shortest path algorithm can be reconfigured to work implicitly on $\overrightarrow{G}_v^D$ when given G, D, and v.

IsLatencyConstrainedCDS(G, D, s**)**:

1. for each $v \in V$ do
   (a) compute shortest paths from v in $\overrightarrow{G}_v^D$;
   (b) if $\text{dist}_{\overrightarrow{G}_v^D}(v,t) > s$ for some $t \in V \setminus \{v\}$, then return "no";
2. return "yes."

Proposition 1. *The algorithm* IsLatencyConstrainedCDS *correctly determines whether a given subset $D \subseteq V$ of vertices is a latency-s CDS for a directed graph $G = (V, E)$:*

- *in $O(mn + n^2 \log\log n)$ time and linear space under nonnegative edge weights;*
- *in $O(mn)$ time and linear space in the hop-based case.*

Here, we are using the algorithm of Thorup (2004) to compute SSSP in time $O(m + n \log\log n)$ in the nonnegative weights case, and BFS to solve SSSP in the hop-based case.

Given that this or some other verification procedure will be called repeatedly in our implementation, it is important that it runs as quickly as possible. For example, we would like to know: is there a different verification procedure that, say, runs in linear time $O(m + n)$? Unfortunately, under a complexity assumption called the strong exponential time hypothesis (SETH) of Impagliazzo et al. (2001) and Impagliazzo and Paturi (2001), this is not possible.

Proposition 2. *If SETH holds, then for every $\varepsilon > 0$ there exists no algorithm for verifying that a subset D of vertices is a latency-s CDS that runs in time $O(m^{2-\varepsilon})$, even in the simplest nontrivial case of hop-based distances and $s = 2$.*

The proof and discussions regarding the limitations of local search are provided in Section 1 of the online supplement.

2.2. The Inapproximability of the Minimum Latency-s CDS Problem

We provide a hardness result for approximating the size of a minimum latency-s CDS. It is based on the hardness result of Dinur and Steurer (2014) that states that approximating the minimum hitting set problem to within a factor of $(1 - \varepsilon)\ln h$ is NP-hard for every $\varepsilon > 0$, where h refers to the number of subsets to hit (cf. Raz and Safra 1997, Alon et al. 2006, Moshkovitz 2015). Also, see similar hardness results based on the stronger assumption that NP does not have quasi-polynomial-time algorithms (Lund and Yannakakis 1994, Feige 1998). Note that $|U| = O(h^c)$ for some constant c in Dinur and Steurer's result.

Problem: The minimum hitting set problem.
Input: a family $F_1, \ldots, F_h \subseteq U$ of subsets of U.
Output: A minimum cardinality subset $D \subseteq U$ such that $|D \cap F_i| \geq 1$ for every $i = 1, \ldots, h$.

Theorem 1. *There exists a polynomial-time algorithm that, when given an instance $((F_1, \ldots, F_h), U)$ of the minimum hitting set problem, creates an instance $(G = (V, E), w, s)$ of the minimum latency-s CDS problem that satisfies:*

- *$|V| = 4 + h + |U|$ and $s = 2 = \text{diam}(G)$ and $w_e = 1$ for each $e \in E$;*
- *there exists a k-hitting set if and only if there exists a $(k + 2)$-vertex latency-s CDS.*

Proof. Let $V = \{r, a, b, c\} \cup T \cup U$, where $T = \{t_1, \ldots, t_h\}$. Thus, $|V| = 4 + h + |U|$. Construct E by bidirecting the following edges. Connect r to every vertex of $U \cup \{a\}$. Connect a to every vertex of U. Connect b to every vertex of $T \cup U$. Make $\{a, b, c\}$ a triangle. Finally, for each F_i in the hitting set instance, connect t_i to every vertex $v \in F_i \subseteq U$.

($\Rightarrow$) Suppose that $D \subseteq U$ is a hitting set of size k. It can be verified that $D \cup \{a, b\}$ is a latency-2 CDS for G, that is, that for every ordered pair of nodes (i, j) with $i \neq j$ and $(i, j) \notin E$, there is a node $v \in D \cup \{a, b\}$ such that (i, v) and (v, j) are edges in E.

($\Leftarrow$) Now, suppose that $D \subseteq V$ is a latency-2 CDS of size $k + 2$. We argue that $D \cap U$ is a hitting set of size at most k. Observe that there is no edge (c, r) and so to ensure 2-hop communication from c to r, D must contain a vertex from $N^+(c) \cap N^-(r)$, and $N^+(c) \cap N^-(r) = \{a\}$ so $a \in D$. Similarly, (c, t_1) is not an edge and $N^+(c) \cap N^-(t_1) = \{b\}$ so $b \in D$. This shows that $|D \cap U| \leq |D| - 2 = k$. Now we show that $D \cap U$ is a hitting set. Recall that, for each $i = 1, \ldots, h$, the edge (r, t_i) does not exist. So, because D is a latency-2 CDS, at least one vertex from $N^+(r) \cap N^-(t_i) = F_i \subseteq U$ must belong to D. Thus, $D \cap U$ is a hitting set of size at most k. □

Corollary 1 (Inapproximability). *There is a constant $\alpha > 0$ such that it is NP-hard to approximate the minimum latency-2 CDS problem to within a factor of $\alpha \ln n$, where n refers to the number of vertices, even under bidirected edges and hop-based distances.*

Proof. This follows by Theorem 1 and the inapproximability of hitting set (Raz and Safra 1997, Alon et al. 2006, Dinur and Steurer 2014, Moshkovitz 2015). □

3. Integer Programming Formulations

In what follows, we propose two IP formulations for the minimum latency-s CDS problem: CUT and POLY.

3.1. Formulation CUT

Here, we propose the formulation called CUT. It has n binary variables and an exponential number of constraints—one for each (minimal) length-s vertex cut.

Definition 4 (Length-s Vertex Cut). A subset $C \subseteq V$ of vertices is a length-s vertex cut of a directed, edge-weighted graph $G = (V, E)$ if $\text{diam}(G - C) > s$.

The correctness of formulation CUT is a consequence of the following characterization.

Proposition 3 (Characterization of Latency-s CDS). *A subset $D \subseteq V$ of vertices is a latency-s CDS for G if and only if $|D \cap C| \geq 1$ for every length-s vertex cut $C \subset V$.*

Proof. ($\Rightarrow$) Assume that $D \subseteq V$ is a latency-s CDS and suppose, for sake of contradiction, that $C \subset V$ is a length-s vertex cut with $|D \cap C| = 0$. By definition of length-s vertex cut, $\text{diam}(G - C) > s$, that is, there exist vertices $a, b \in V \setminus C$ such that $\text{dist}_{G-C}(a, b) > s$. By assumption that D is a latency-s CDS, there is an a-b path of length at most s whose interior vertices belong solely to D, that is, $\text{dist}_{G[D\cup\{a,b\}]}(a, b) \leq s$. Because $D \cup \{a, b\} \subseteq V \setminus C$, we have $\text{dist}_{G[V\setminus C]}(a, b) \leq \text{dist}_{G[D\cup\{a,b\}]}(a, b)$ which results in the following contradiction:

$$s < \operatorname*{dist}_{G-C}(a, b) \triangleq \operatorname*{dist}_{G[V\setminus C]}(a, b) \leq \operatorname*{dist}_{G[D\cup\{a,b\}]}(a, b) \leq s.$$

($\Leftarrow$) By the contrapositive. Suppose that $D \subseteq V$ is not a latency-s CDS, that is, there exist vertices $a, b \in V$ such that there is no a-b path of length at most s whose interior vertices belong to D, that is, $\text{dist}_{G[D\cup\{a,b\}]}(a, b) > s$. This implies that $\text{diam}(G[D \cup \{a, b\}]) > s$, and so $C := V \setminus (D \cup \{a, b\})$ is a length-s vertex cut. Moreover, $|D \cap C| = 0$, as desired. □

Proposition 3 immediately implies the correctness of the formulation CUT:

$$\min \sum_{i\in V} x_i \tag{8}$$

$$\sum_{i\in C} x_i \geq 1, \quad \forall \text{ length-}s \text{ vertex cut } C \subset V \tag{9}$$

$$x_i \in \{0, 1\}, \quad \forall i \in V. \tag{10}$$

In general, there can be exponentially many of the constraints (9), even if we restrict ourselves to *inclusion-minimal* length-s vertex cuts. This formulation generalizes the CDS formulation based on vertex cuts that is essentially due to Fujie (2004). We address the separation complexity for constraints (9) in Section 4.

Not every valid inequality of the form $\sum_{i\in C} x_i \geq 1$ is a length-s vertex cut inequality. For example, $\sum_{i\in V\setminus\{v\}} x_i \geq 1$ is valid when $G = (V, E)$ is the bidirected 4-cycle, but $V \setminus \{v\}$ is not a length-s vertex cut. However, the following shows that the length-s vertex cut inequalities are the only *meaningful* valid inequalities of this type.

Lemma 1. *Let $C \subset V$. The inequality $|S \cap C| \geq 1$ holds for every latency-s CDS $S \subseteq V$ if and only if C is a superset of some length-s vertex cut C'.*

Proof. The 'if' direction follows easily by Proposition 3, so suppose that $|S \cap C| \geq 1$ holds for every latency-s CDS $S \subseteq V$. Let $D = V \setminus C$. By our assumption, D cannot be a latency-s CDS, that is, there exist vertices $a, b \in V$ such that $\text{dist}_{G[D\cup\{a,b\}]}(a, b) > s$. So, $s < \text{diam}(G[D \cup \{a, b\}]) = \text{diam}(G - C')$, where $C' = V \setminus (D \cup \{a, b\})$. Thus, C' is a length-s vertex cut for G, and $C \supseteq C'$, as desired. □

Given that the minimum latency-s CDS problem admits the formulation CUT (and by Lemma 1), there are immediate polyhedral consequences (cf. Sassano 1989), so we provide the following proposition without proof.

Proposition 4 (Basic Polyhedral Analysis). *The convex hull of (characteristic vectors of) latency-s CDS is full dimensional if and only if every length-s vertex cut has size at least two. Further, if it is full dimensional, then*

1. *for each $v \in V$,*
   (a) *$x_v \leq 1$ induces a facet, and*
   (b) *$x_v \geq 0$ induces a facet if and only if v does not belong to a length-s vertex cut of size two;*
2. *for $C \subset V$, the inequality $\sum_{i\in C} x_i \geq 1$ induces a facet if and only if*
   (a) *C is a minimal length-s vertex cut, and*
   (b) *for each $v \in V \setminus C$ there exists $c \in C$ such that $((V \setminus C) \cup \{c\}) \setminus \{v\}$ is a latency-s CDS.*

3.2. Generalizing the Formulation CUT for Fault-Tolerance

Here, we consider the *robust* or *fault-tolerant* variant of a latency-s CDS. That is, we are interested in a vertex subset that remains a latency-s CDS when few vertices fail.

Definition 5 (r-Robust Latency-s CDS). A subset $D \subseteq V$ of vertices is an r-robust latency-s CDS for graph G if, for every $F \subseteq D$ with $|F| < r$, the vertex subset $D \setminus F$ is a latency-s CDS for G.

A consequence of Proposition 3 is the following characterization.

Corollary 2 (Characterization of r-Robust Latency-s CDS). *A subset $D \subseteq V$ of vertices is an r-robust latency-s CDS if and only if $|D \cap C| \geq r$ for every length-s vertex cut $C \subset V$.*

Corollary 2 immediately implies the correctness of the following formulation for the minimum r-robust latency-s CDS problem:

$$\min \sum_{i\in V} x_i \tag{11}$$

$$\sum_{i\in C} x_i \geq r, \quad \forall \text{ length-}s \text{ vertex cut } C \subset V \tag{12}$$

$$x_i \in \{0, 1\}, \quad \forall i \in V. \tag{13}$$

This formulation generalizes previously existing formulations for the minimum k-k-CDS problem (Ahn and Park 2015, Buchanan et al. 2015).

Figure 1(b) gives a feasible solution to this problem when $(s, r) = (3, 2)$ (using hop-based distances

and treating the undirected edges as bidirected edges). That is, the gray vertices remain a latency-3 CDS when one of them fails. This is also an optimal solution for $(s,r) = (2,1)$. However, there is no solution for $(s,r) = (2,2)$, as evidenced by the length-2 vertex cut $C = \{5\}$. Indeed, this implies that the inequality $x_5 \geq 2$ is valid, but of course no binary vector x can satisfy this constraint.

We remark that our formalization of a fault-tolerant low-latency virtual backbone might *not* provide for r vertex-disjoint paths of length at most s (of CDS vertices) between every pair of vertices. An example is given in Figure 3, treating the undirected edges as bidirected edges. This should not be surprising given that there is, in general, no "Menger's theorem" for length-bounded paths (cf. Lovász et al. 1978).

3.3. Formulation POLY

Here, we propose the formulation called POLY. It is introduced primarily for comparison purposes and is inspired by a formulation for s-clubs given by Veremyev and Boginski (2012). It applies to the hop-based case.

As before, the binary variable x_i represents the decision to include vertex i in the latency-s CDS. The binary variable y^t_{ij} equals one if and only if there exists a directed path in G from i to j of length exactly t whose interior vertices belong to the chosen CDS. This variable is only defined when $t \geq 2$ and should not be confused with y_{ij} raised to the tth power. To formulate our problem, we should write constraints that impose the following condition:

$$y^t_{ik} = 1 \iff \big(\text{there exists } j \in N^-(k) \text{ such that } y^{t-1}_{ij} = 1 \text{ and } x_j = 1\big).$$

In words, there is a path (across CDS nodes) from i to k of length t if and only if (a) there is a path (across CDS nodes) of length $t-1$ from i to some in-neighbor j of node k and (b) node j belongs to the CDS. When $t \geq 3$, this equivalence can be formulated as follows.

$$(\Leftarrow) \quad y^{t-1}_{ij} + x_j \leq y^t_{ik} + 1 \quad \forall j \in N^-(k)$$
$$(\Rightarrow) \quad y^t_{ik} \leq \sum_{j \in N^-(k)} y^{t-1}_{ij} x_j.$$

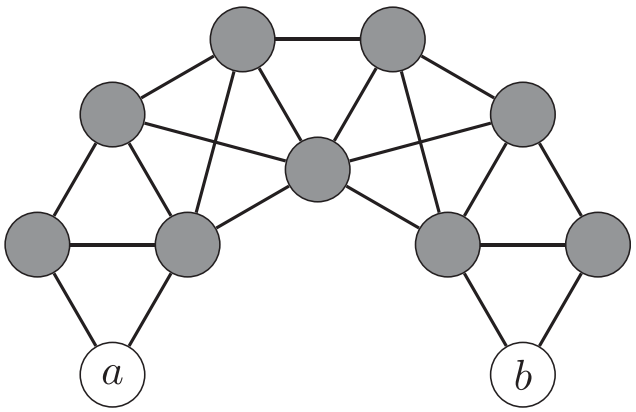
The second implication is enforced via a constraint that has products of binary variables. For linearization purposes, introduce (binary) variables z^{t-1}_{ij} to replace the terms $y^{t-1}_{ij} x_j$. To impose that $z^{t-1}_{ij} = y^{t-1}_{ij} x_j$, use the usual linear constraints:

$$z^{t-1}_{ij} \leq y^{t-1}_{ij}$$
$$z^{t-1}_{ij} \leq x_j$$
$$y^{t-1}_{ij} + x_j \leq z^{t-1}_{ij} + 1.$$

These ideas lead to the following formulation, where the special case $t = 2$ is handled via constraints (15) and (16). Let $T_{\geq 3} := \{3, \ldots, s\}$ and $N^-[j] := N^-(j) \cup \{j\}$.

$$\min \sum_{i \in V} x_i \tag{14}$$
$$x_j \leq y^2_{ik} \qquad j \in N^+(i) \cap N^-(k),\ i \in V \setminus \{k\},\ k \in V \tag{15}$$
$$y^2_{ik} \leq \sum_{j \in N^+(i) \cap N^-(k)} x_j \qquad i \in V \setminus \{k\},\ k \in V \tag{16}$$
$$y^{t-1}_{ij} + x_j \leq y^t_{ik} + 1 \qquad i \in V \setminus \{j,k\},\ (j,k) \in E,\ t \in T_{\geq 3} \tag{17}$$
$$y^t_{ik} \leq \sum_{j \in N^-(k)} z^{t-1}_{ij} \qquad i \in V \setminus \{k\},\ k \in V,\ t \in T_{\geq 3} \tag{18}$$
$$z^{t-1}_{ij} \leq y^{t-1}_{ij} \qquad i \in V \setminus \{j\},\ j \in V,\ t \in T_{\geq 3} \tag{19}$$
$$z^{t-1}_{ij} \leq x_j \qquad i \in V \setminus \{j\},\ j \in V,\ t \in T_{\geq 3} \tag{20}$$
$$y^{t-1}_{ij} + x_j \leq z^{t-1}_{ij} + 1 \qquad i \in V \setminus \{j\},\ j \in V,\ t \in T_{\geq 3} \tag{21}$$
$$\sum_{t=2}^{s} y^t_{ij} \geq 1 \qquad i \in V \setminus N^-[j],\ j \in V \tag{22}$$
$$x_i \in \{0,1\} \qquad i \in V \tag{23}$$
$$y^t_{ij} \in \{0,1\} \qquad i \in V \setminus \{j\},\ j \in V,\ t \in \{2, \ldots, s\} \tag{24}$$
$$z^t_{ij} \in \{0,1\} \qquad i \in V \setminus \{j\},\ j \in V,\ t \in \{2, \ldots, s-1\}. \tag{25}$$

Figure 3. Lack of Menger's Property



Notes. Observe that this is a unit disk graph. A CDS that maintains 5-hop communication paths when any one vertex fails but that does not have a pair of vertex-disjoint length-5 a-b paths.

The constraints (22) ensure that there is a path of length at most s (across CDS vertices) from i to j when $(i,j) \notin E$. So, by the ideas presented above, it is straightforward to prove the following.

Theorem 2. *Under hop-based distances, the above is a correct formulation for the minimum latency-s CDS problem and has $\Theta(sn^2)$ variables, $\Theta(snm)$ constraints, and $\Theta(snm)$ nonzeros.*

Because modern MIP solvers use sparse matrix representation, this formulation's size in computer memory can be approximated by the number $\Theta(snm)$ of nonzeros. This is much less than the quantity obtained by multiplying the number of variables by the number of constraints.

Not all of these variables and constraints may be necessary. For example, if G is bidirected, we can assume that $y^t_{ij} = y^t_{ji}$. If desired, the user can impose these constraints $y^t_{ij} = y^t_{ji}$ when implementing the formulation, and the MIP solver will perform the appropriate substitutions in its presolve phase.

Based on our computational experiments, it is possible that formulation POLY is weaker than CUT, although we could not find a proof. In Section 2 of the online supplement, we provide a fractional point (x^*, y^*, z^*) that belongs to POLY's LP relaxation, but its x^* does not belong to CUT's LP relaxation.

4. The Complexity of the Formulations

In this section, we determine the separation complexity for the constraints defining formulation CUT and its fault-tolerant generalization. On the way, we answer an open question of Xu et al. (2005) regarding the complexity of computing a graph's fault diameter.

4.1. Computing the Fault Diameter of Graphs

As a helpful first step to determining the separation complexity, we show that a related problem, which we call Diameter Interdiction by Node Deletion, is NP-complete.

Problem: Diameter Interdiction By Node Deletion.
Input: a simple graph $G = (V, E)$ and integers q and L.
Question: Is there a subset $C \subset V$ of q vertices such that $\text{diam}(G - C) > L$?

This problem is defined for an undirected and unweighted graph G, and the diameter that is referred to is hop-based.

Theorem 3. *For each $L \geq 5$, Diameter Interdiction by Node Deletion is NP-complete.*

To prove this theorem, we craft reductions from Length-Bounded a-b Node Cut, which is known to be NP-complete and hard to approximate (Baier et al. 2010). Notice that this problem has specified end nodes a and b, while Diameter Interdiction by Node Deletion does not. We provide two reductions, given in Lemmata 2 and 3, which, together, prove Theorem 3.

Problem: Length-Bounded a-b Node Cut.
Input: A simple graph $G' = (V', E')$, nonadjacent $a, b \in V'$, and integers q' and L'.
Question: Is there a subset $C' \subseteq V' \setminus \{a, b\}$ of q' vertices such that $\text{dist}_{G'-C'}(a,b) > L'$?

Lemma 2. *For each odd $L \geq 5$, Diameter Interdiction by Node Deletion is NP-complete.*

Proof. Membership in NP is obvious. For the reduction, consider an instance of Length-Bounded a-b Node Cut defined by graph $G' = (V', E')$, vertices $a, b \in V'$, and integers q' and odd $L' \geq 5$. Let $q = q'$ and $L = L'$. Now we construct $G = (V, E)$. The idea is to connect every pair of vertices from G' (besides a and b) by carefully adding many short paths so that the only possible way to cheaply disrupt the diameter of G is to cut all short paths from a to b. Construct V as follows.

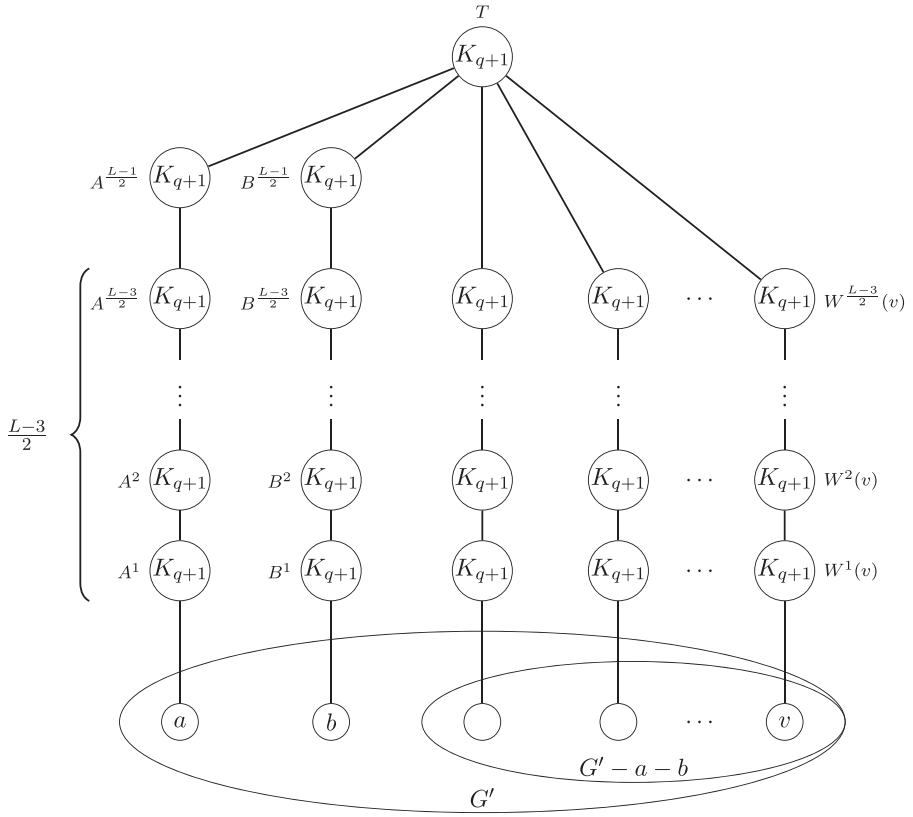
$$V := V' \cup T \cup A \cup B \cup W$$
$$T := \{t_i \mid 1 \leq i \leq q+1\}$$
$$A := \left\{a^j_i \,\middle|\, 1 \leq i \leq q+1,\ 1 \leq j \leq \frac{L-1}{2}\right\}$$
$$B := \left\{b^j_i \,\middle|\, 1 \leq i \leq q+1,\ 1 \leq j \leq \frac{L-1}{2}\right\}$$
$$W := \left\{v^j_i \,\middle|\, 1 \leq i \leq q+1,\ 1 \leq j \leq \frac{L-3}{2}, v \in V' \setminus \{a,b\}\right\}.$$

Notice that $|V| = O(qL|V'|)$ and $q \leq |V'|$ and $L \leq |V'|$, so the reduction will be polynomial.

Construct the edge set E of G as follows. First, connect the vertices of V' so that $G[V'] = G'$. Then make T a clique in G. Similarly, make each $A^j := \{a^j_i \mid 1 \leq i \leq q+1\}$ a clique. Do the same for each $B^j := \{b^j_i \mid 1 \leq i \leq q+1\}$ and for each $W^j(v) := \{v^j_i \mid 1 \leq i \leq q+1\}$. Connect a to every vertex of A^1; and every vertex of A^1 to every vertex of A^2; and so on. Connect b to every vertex of B^1; and every vertex of B^1 to every vertex of B^2; and so on. Then for every $v \in V' \setminus \{a,b\}$, connect v to every vertex of $W^1(v)$; and every vertex of $W^1(v)$ to every vertex of $W^2(v)$; and so on. Finally, letting $p = \frac{L-1}{2}$, connect every vertex of T to every vertex of $A^p \cup B^p \cup (\cup_{v \in V' \setminus \{a,b\}} W^{p-1}(v))$. Creating E obviously can be done in polynomial time. See Figure 4 for an illustration.

Observe that there exist at least $q+1$ (internally) node-disjoint paths of length at most L between every pair of vertices of G (the interior vertices of which belong to $V \setminus V'$), except possibly for the pair $\{a, b\}$. Moreover, (simple) a-b paths of length at most L in G can only cross vertices of V'. Thus, it can be argued that the instance (G', a, b, q', L') of Length-Bounded a-b Node Cut is a "yes" if and only if the instance (G, q, L) of Diameter Interdiction by Node Deletion is a "yes." □

Figure 4. Illustration of the Reduction for Odd $L \geq 5$



Note. Here, K_n is a complete graph on n nodes.

Lemma 3. *For each even $L \geq 5$,* Diameter Interdiction by Node Deletion *is NP-complete.*

Proof. Membership in NP is obvious. For the reduction, consider an instance of Length-Bounded a-b Node Cut defined by graph $G' = (V', E')$, vertices $a, b \in V'$, and integers q' and even $L' \geq 5$. Let $q = q'$ and $L = L'$. Now we construct $G = (V, E)$. The main idea behind the reduction is the same as before, but the construction is slightly different. Construct V as follows.

$$V := V' \cup T \cup T' \cup W$$
$$T := \left\{ t_i \,\middle|\, 1 \leq i \leq q+1 \right\}$$
$$T' := \left\{ t'_i \,\middle|\, 1 \leq i \leq q+1 \right\}$$
$$W := \left\{ v_i^j \,\middle|\, 1 \leq i \leq q+1,\ 1 \leq j \leq \frac{L}{2} - 1,\ v \in V' \right\}.$$

Notice that $|V| = O(qL|V'|)$ and $q \leq |V'|$ and $L \leq |V'|$, so the reduction will be polynomial.

Construct the edge set E of G as follows. First, connect the vertices of V' so that $G[V'] = G'$. Then make $T \cup T'$ a clique in G. Similarly, make each $W^j(v) := \{v_i^j \mid 1 \leq i \leq q+1\}$ a clique. For every $v \in V'$, connect v to every vertex of $W^1(v)$; and every vertex of $W^1(v)$ to every vertex of $W^2(v)$; and so on. Let $p = \frac{L}{2} - 1$. Connect every vertex of T to every vertex of $\cup_{v \in V' \setminus \{b\}} W^p(v)$. Similarly, connect every vertex of T' to every vertex of $\cup_{v \in V' \setminus \{a\}} W^p(v)$. Creating E obviously can be done in polynomial time. See Figure 5 for an illustration.

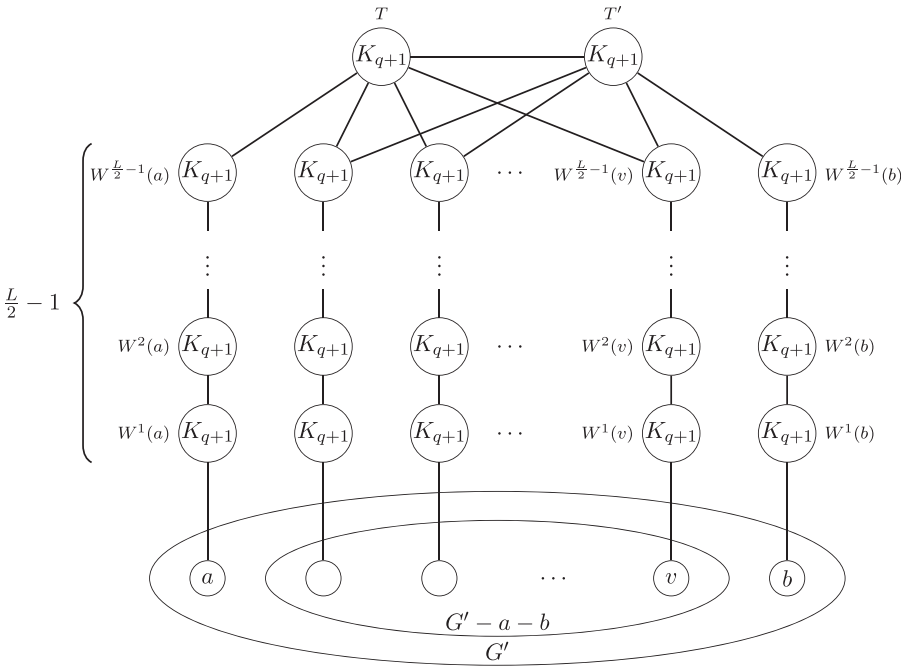
Observe that there exist at least $q+1$ (internally) node-disjoint paths of length at most L between every pair of vertices of G (the interior vertices of which belong to $V \setminus V'$), except possibly for the pair $\{a, b\}$. Moreover, (simple) a-b paths of length at most L in G can only cross vertices of V'. Thus, one can argue that the instance (G', a, b, q', L') of Length-Bounded a-b Node Cut is a "yes" if and only if the instance (G, q, L) of Diameter Interdiction by Node Deletion is a "yes." □

Researchers have studied related notions of the *fault diameter* of a graph (Krishnamoorthy and Krishnamurthy 1987, Xu 2001). For example, Xu (2001) defines the f-fault diameter of graph $G = (V, E)$ to be

$$D_f(G) := \max\{\operatorname{diam}(G - F) \mid F \subseteq V,\ |F| < f\},$$

and states that computing this value "is a quite difficult problem," but no justification is given.[3] Later, Xu et al. (2005) listed its NP-hardness as an open problem. Theorem 3 implies that computing $D_f(G)$ is indeed NP-hard when f is part of the input, say, by letting $f = q + 1$ and returning "yes" if $D_f(G) > L$.

Figure 5. Illustration of the Reduction for Even $L \geq 5$



Corollary 3. *Computing the f-fault diameter is NP-hard when f is part of the input.*

4.2. The Separation Problem for CUT

Formulation CUT has an exponential number of constraints (9), as does its fault-tolerant generalization (12), making it a nontrivial question as to how they should be used. A helpful observation, however, is that by the polynomial equivalence of optimization and separation (Grötschel et al. 1993), their LP relaxations can be solved in polynomial time if and only if their separation problems (defined below) can be solved in polynomial time.

Problem: Separation Problem for Formulation CUT.

Input: a directed and edge-weighted graph $G = (V, E)$, a weight $x^*_v \in [0, 1]$ for each $v \in V$, an integer $r \geq 1$, a number s.

Output: (if any exist) a length-s vertex cut $C \subseteq V$ with $\sum_{i \in C} x^*_i < r$.

For purposes of generality, we define this separation problem for the fault-tolerant generalization, which has right-hand-side r. We provide both positive and negative results.

Theorem 4. *Under hop-based distances, the separation problem is:*

1. *polynomial-time solvable for $s \in \{2, 3, 4\}$, for every $r \geq 1$;*
2. *(in its decision version) NP-complete for every $s \geq 5$, even when $r = 1$.*

Proof. First, we prove that item 2 holds. Membership in NP is clear, so we only show hardness. The reduction is from an instance of Diameter Interdiction by Node Deletion given by (G, q, L), which is NP-complete for each $L \geq 5$ by Theorem 3. Bidirect $G = (V, E)$ yielding directed graph $\overleftrightarrow{G} = (V, \overleftrightarrow{E})$. Let $r = 1$, $s = L$, and $x^*_i = \frac{1}{q+1}$ for every $i \in V$. We argue that (G, q, L) is a "yes" instance of Diameter Interdiction by Node Deletion if and only if $(\overleftrightarrow{G}, x^*, r, s)$ admits a violated length-s vertex cut inequality (12). Suppose there is a violated length-s vertex cut inequality (12) for some $C \subseteq V$. Then $\frac{|C|}{q+1} = \sum_{i \in C} x^*_i < 1$, that is, $|C| \leq q$, and the instance of Diameter Interdiction by Node Deletion is a "yes." Now, if there is a length-s vertex cut $C' \subseteq V$ with $|C'| \leq q$ for $\overleftrightarrow{G}$, then $\sum_{i \in C'} x^*_i = \frac{|C'|}{q+1} \leq \frac{q}{q+1} < 1$ and so x^* violates the length-s vertex cut inequality $\sum_{i \in C'} x_i \geq 1$.

Now, we discuss *why* item 1 holds. In the cases $s \in \{2, 3, 4\}$, we can find a *most-violated* length-s vertex cut inequality (12) by computing, for each $(a, b) \in (V \times V) \setminus E$, a minimum-weight length-s a,b-vertex cut and comparing its weight to r. The cases $s \in \{2, 3\}$ are fairly straightforward; for example, for $s = 2$ the solution is $N^+(a) \cap N^-(b)$. The case $s = 4$ was (essentially) shown by Lovász et al. (1978) to be polynomial-time solvable by reducing it to a particular instance of the min-cut problem (cf. theorem 4.3.1 of Xu (2001)). Because this min-cut instance can be constructed in linear time (and it is actually a subgraph of the input graph), this minimum-weight length-s a,b-vertex cut subproblem can be solved in time $O(mn)$ by Orlin (2013). Solving these subproblems for every missing edge (a, b) gives a total time of $O(mn^3)$, which is polynomial. □

By standard arguments, the flow-based separation routines referenced in the proof of Theorem 4 imply polynomial-size extended formulations for the LP

relaxation of CUT when $s \in \{3, 4\}$ (see Martin 1991). However, these formulations would have roughly mn^2 variables, making them too large to be practical. Hence, we do not discuss them further.

4.3. Verification and Integer Separation for the Fault-Tolerant Variant

The problem of verifying whether a given subset $D \subseteq V$ of vertices is an r-robust latency-s CDS is nontrivial. In a brute force approach, enumerate all subsets $F \subseteq D$ of $r-1$ vertices and verify that $D \setminus F$ is indeed a latency-s CDS. By algorithm IsLatencyConstrainedCDS, this takes time $\binom{|D|}{r-1}O(n^3) = O(n^{r+2})$, which is polynomial for any constant r. A natural question is whether this test can be performed in polynomial time when r is part of the problem input. Unfortunately, the likely answer is "no," as this is coNP-complete.

Corollary 4. *When r is part of the input, the problem of verifying whether $D \subseteq V$ is an r-robust latency-s CDS is coNP-complete for each fixed $s \geq 5$. This holds even for bidirected edges and hop-based distances.*

Proof. Membership in coNP follows because a length-s vertex cut $C \subseteq V$ with $|C| < r$ is a suitable witness when it is a "no" instance. For the reduction, consider an instance of Diameter Interdiction by Node Deletion defined by a simple graph $G = (V, E)$ and integers q and L. Bidirect its edges and let $s = L$, $r = q + 1$, and $D = V$. It can be observed that the instance of Diameter Interdiction by Node Deletion is a "yes" instance if and only if D is *not* an r-robust latency-s CDS of this bidirected graph. □

Remark 1. As a consequence of Corollary 4, the separation problem for the constraints (12), with r being part of the input, is hard *even when x^* is integer*.

5. Computational Experiments

In this section, we provide results from our computational experiments. First, we demonstrate the importance of (quickly) strengthening the length-s vertex cut inequalities. Second, we provide computational results demonstrating the importance of providing an initial heuristic solution to the MIP solver. Third, we compare our full implementation of CUT with the polynomial-size formulation POLY. Our tests demonstrate the superiority of CUT over POLY. Finally, we experiment with formulation CUT for:

1. $s \in \{\text{diam}(G), \text{diam}(G)+1, \text{diam}(G)+2, n-1\}$;
2. the fault-tolerant case with $r = 2$;
3. a class of instances representing node-weighted, transmitter-based delays.

All of our experiments are conducted on a Dell Precision Tower 7000 Series (7810) machine running Windows 10 enterprise, x64, with Intel® Xeon® Processor E52630 (v.4) (10 cores, 2.2GHz, 3.1GHz Turbo, 2133MHz, 25MB, 85W)—that is, 20 logical processors—and 32 GB memory. The IP formulations were implemented in Microsoft Visual Studio 2015 in C++ for Gurobi (v.7.0.2). We use default settings with the exception that we force Gurobi to use the concurrent method (which uses primal simplex, dual simplex, and barrier on different threads) for solving the root LP relaxation for POLY, as this formulation is highly degenerate and typically barrier is fastest. We impose a time limit of 3600 seconds on each instance and use the same test instances that have been used in the previous literature on the minimum CDS problem by Lucena et al. (2010), Simonetti et al. (2011), Fan and Watson (2012), Gendron et al. (2014), Buchanan et al. (2015), and Li and Aneja (2017). This testbed includes both real-life and synthetically generated instances, all of which are undirected. In our experiments, we bidirect their edges. The test instances and code are available at https://github.com/hamidrezavalidi/LCDS.

5.1. Importance of Strengthening the Inequalities

Because formulation CUT can have exponentially many constraints when $s \geq 3$, we initialize it with only some of the constraints. Others are added as needed via Gurobi's lazy constraint callback features. Specifically, we start with the vertex cuts given by $N^+(i)$, $i \in V$ (or inclusion-minimal subsets thereof that are also length-s vertex cuts). Then, within the branch-and-bound tree, violated length-s vertex cut inequalities are added on-the-fly.

Because the separation problem for the length-s vertex cut inequalities is NP-hard, we only separate *integer* points that the solver encounters. Each of these possible solutions $D \subseteq V$ will satisfy the initial constraints given to the solver, but D may not actually be feasible for the latency-s CDS problem. In this case, $C := V \setminus D$ is a length-s vertex cut for the graph, and the inequality $\sum_{i \in C} x_i \geq 1$ would be valid for our problem and would cut off the binary point representing D. However, this inequality is likely very weak, so we strengthen the inequality, that is, find a minimal subset of C that is also a length-s vertex cut. This is done when initializing the formulation with the vertex cuts $N^+(i)$, $i \in V$ and also when adding inequalities on-the-fly. The details are given in Section 3 of the online supplement. Theorem 2 of the online supplement shows that one can find a violated minimal length-s vertex cut inequality in time $O(n^3)$.

We also experimented with separating fractional points, particularly when $s = 3$ as this case of the separation problem is polynomial-time solvable. However, the fastest separation procedure that we are aware of takes time $O(mn^3)$ and was ultimately unhelpful—in all nine of the different implementations that we tried. See Section 4 of the online supplement for more details.

5.2. The Importance of Providing a Heuristic Solution to the Solver

See Section 5 of the online supplement.

5.3. Comparison with Formulation POLY

In Table 1, we compare the performance of CUT with that of POLY. In these tests, we set $s = \text{diam}(G)$, provide an MIP start using BestInHeuristic, and exclude instances with $s = 2$. The reason for excluding the $s = 2$ comparisons is that CUT and POLY are equally strong when $s = 2$, and so CUT will obviously perform better due to its smaller size. Because our instances are bidirected, we fix $y^t_{ij} = y^t_{ji}$ as discussed in Section 3.3.

The results demonstrate the superiority of CUT. It solves the 11 instances solved by POLY and 9 others. Ten instances are left unsolved by both approaches; CUT provides better bounds on all of them. The formulation CUT also *quickly* solves some instances that POLY left unsolved after an hour. For example, CUT solved v50_d5, v70_d5, v70_d10, and v70_d30 each in under 5 seconds, whereas POLY solved none of them in the time limit.

Table 1. A Comparison of the Performance of Formulation CUT with That of POLY

			POLY		CUT	
Graph	s	H Obj	Obj	Total	Obj	Total
v30_d10	8	15	15	2.71	15	0.02
v30_d20	5	8	8	111.14	8	0.03
v30_d30	3	8	8	12.59	8	0.10
v50_d5	14	32	[31,32]	—	32	0.07
v50_d10	5	20	18	2,711.41	18	0.18
v50_d20	3	14	14	6.31	14	0.11
v50_d30	3	8	8	398.41	8	1.12
v70_d5	8	36	[26,36]	—	32	0.53
v70_d10	4	31	[28,29]	—	29	2.98
v70_d20	3	18	[11,18]	—	17	305.96
v70_d30	3	8	[6,7]	—	7	3.33
v100_d5	5	57	[42,57]	—	56	17.91
v100_d10	4	31	[13,31]	—	[22,26]	—
v100_d20	3	20	[9,20]	—	[14,20]	—
v120_d5	6	40	[16,40]	—	31	1,087.17
v120_d10	3	68	63	1,522.53	63	10.31
v120_d20	3	21	[7,21]	—	[10,21]	—
v120_d30	3	12	[4,12]	—	[7,12]	—
v150_d5	5	54	[19,54]	—	[30,54]	—
v150_d10	3	65	[35,65]	—	[43,61]	—
v150_d20	3	22	[6,22]	—	[9,22]	—
v200_d5	4	92	[43,92]	—	[49,92]	—
v200_d10	3	64	[24,64]	—	[26,64]	—
v200_d20	3	22	[6,22]	—	[8,22]	—
IEEE-14	5	5	5	0.08	5	0.01
IEEE-30	6	14	14	0.29	14	0.01
IEEE-57	12	35	35	57.65	35	0.04
RTS-96	13	40	[35,39]	—	37	0.14
IEEE-118	14	48	48	130.70	48	0.15
IEEE-300	24	139	[6,139]	—	135	11.98

Notes. For all graphs G, we set $s = \text{diam}(G)$ and exclude instances in which $s = 2$. We report the heuristic objective value under the column labeled "H Obj." We report the optimal objective (or the best lower/upper bounds $[L, U]$ after 1 hour) under the columns labeled "Obj." We also give the total solve time (Total), where a dash indicates > 3,600 seconds.

5.4. The Cost of Low Latency

Imposing that a dominating set be *connected* is not too costly. Indeed, the domination number $\gamma(G)$ and the connected domination number $\gamma_c(G)$ are a constant factor apart. Specifically, they satisfy $\gamma(G) \le \gamma_c(G) \le 3\gamma(G) - 2$ (see Haynes et al. 1998). In contrast, we show that the cost of low latency can be very large—even when decreasing the latency parameter s by one. We denote by $\gamma^{lat}_s(G)$ the size of a minimum latency-s CDS in G.

Proposition 5. (Potentially large cost of low latency). *For every latency parameter $s \ge 2$, there is an infinite class of graphs G for which $\gamma^{lat}_{s+1}(G) \le s$, but $\gamma^{lat}_s(G) \ge \Omega(n)$. This holds even when edges are bidirected and distances are hop-based.*

Proof. One such class of graphs are obtained by taking the Cartesian products $K_q \square P_s$ of a complete graph K_q and a path graph P_s and then bidirecting the edges. These graphs have sq vertices and diameter s when $q \ge 2$. These graphs $K_q \square P_s$ can be defined as having vertex set $V^1 \cup \cdots \cup V^s$, where each $V^i = \{v^i_1, \ldots, v^i_q\}$. For the edges, let each V^i be a clique, and connect each vertex v^i_j to its counterpart v^{i+1}_j from the next V^{i+1}. Bidirect all edges.

See that $\gamma^{lat}_{s+1}(K_q \square P_s) \le s$, because the s vertices v^i_1 form a feasible solution. Now we show that $\gamma^{lat}_s(K_q \square P_s) \ge \Omega(n)$ in two cases. When $s = 2$, the q vertex subsets $\{v^1_i, v^2_{i+1}\}$ for $i = 1, \ldots, q-1$ and $\{v^1_q, v^2_1\}$ form length-2 cuts and are disjoint. Thus, $\gamma^{lat}_2(K_q \square P_2) \ge q = n/2$. When $s \ge 3$, each vertex v^i_j with $2 \le i \le s-1$ is a length-s vertex cut on its own (by the resultant distance between nodes v^1_j and v^s_j), so $\gamma^{lat}_s(K_q \square P_s) \ge (s-2)q \ge n/3$. □

We observe the cost of low latency "in practice" through the computational results given in Table 2. We report the solution sizes and runtimes for different values of the latency parameter $s \in \{\text{diam}, \text{diam}+1, \text{diam}+2, n-1\}$ under hop-based distances. Thus, we have the strictest case of $s = \text{diam}$ and the most relaxed value of $s = n - 1$, which corresponds to the minimum CDS problem when edges are bidirected. The solve times tend to improve as s increases, and we are able to solve all instances when $s = n - 1$. However, this is not universally the case, for example, for graph v100_d5. Some of the lower bounds can be immediately improved based on the table. For example, we can claim a lower bound of 10 for the instance v150_d20 when $s = \text{diam}$, because 10 is optimal for the less-restrictive case $s = \text{diam} + 1$.

The runtimes for the case $s = n - 1$ closely resemble those given by Buchanan et al. (2015) for the minimum

Table 2. Results for Different Values of the Latency Parameter s

		s = diam			s = diam + 1			s = diam + 2			$s = n - 1$	
Graph	diam	Club	Obj	Total	Club	Obj	Total	Club	Obj	Total	Obj	Total
v30_d10	8	∞	15	0.02	15	15	0.02	15	15	0.02	15	0.04
v30_d20	5	8	8	0.03	7	7	0.01	7	7	0.02	7	0.01
v30_d30	3	∞	8	0.10	5	5	0.05	4	4	0.01	4	0.01
v30_d50	2		7	0.01	3	3	0.01	3	3	0.01	3	0.01
v30_d70	2		3	0.04	2	2	0.01	2	2	0.01	2	0.01
v50_d5	14	32	32	0.07	32	32	0.13	31	31	0.19	31	0.36
v50_d10	5	19	18	0.18	14	14	0.18	13	13	0.11	12	0.23
v50_d20	3	∞	14	0.11	7	7	0.27	7	7	0.17	7	0.17
v50_d30	3	∞	8	1.12	5	5	0.10	5	5	0.12	5	0.12
v50_d50	2		9	0.17	3	3	0.10	3	3	0.02	3	0.02
v50_d70	2		4	0.79	2	2	0.03	2	2	0.02	2	0.02
v70_d5	8	32	32	0.53	29	29	0.74	28	28	1.38	27	0.76
v70_d10	4	∞	29	2.98	17	16	8.87	13	13	0.47	13	0.10
v70_d20	3	∞	17	305.96	8	8	0.21	7	7	0.19	7	0.14
v70_d30	3	∞	7	3.33	5	5	0.17	5	5	0.16	5	0.16
v70_d50	2		10	0.56	3	3	0.05	3	3	0.05	3	0.05
v70_d70	2		5	1.57	2	2	0.10	2	2	0.06	2	0.06
v100_d5	5	∞	56	17.91	40	[34,36]	—	29	29	2,018.43	24	0.56
v100_d10	4	∞	[22,26]	—	15	15	4.24	14	14	0.41	13	0.25
v100_d20	3	∞	[14,20]	—	9	9	2.35	8	8	0.53	8	0.51
v100_d30	2		39	5.00	∞	[7,12]	—	6	6	0.94	6	0.98
v100_d50	2		12	61.27	4	4	0.87	4	4	0.89	4	0.93
v100_d70	2		5	8.17	3	3	1.20	3	3	1.18	3	1.20
v120_d5	6	31	31	1,087.17	28	28	97.26	26	26	4.36	25	0.62
v120_d10	3	∞	63	10.31	∞	[17,31]	—	15	15	202.51	13	0.69
v120_d20	3	∞	[10,21]	—	9	9	5.67	8	8	3.46	8	1.54
v120_d30	3	∞	[7,12]	—	6	6	1.10	6	6	1.20	6	1.10
v120_d50	2		[11,12]	—	4	4	4.78	4	4	3.71	4	3.63
v120_d70	2		5	29.67	3	3	2.06	3	3	2.16	3	2.08
v150_d5	5	∞	[30,54]	—	[28,33]	[26,40]	—	[26,28]	[26,33]	—	26	2.10
v150_d10	3	∞	[43,61]	—	∞	[14,28]	—	16	[15,18]	—	14	4.94
v150_d20	3	∞	[9,22]	—	10	10	379.34	9	9	7.13	9	6.88
v150_d30	2		[35,41]	—	∞	[6,11]	—	6	6	7.65	6	3.96
v150_d50	2		[9,13]	—	4	4	2.38	4	4	2.49	4	2.77
v150_d70	2		6	569.09	3	3	3.29	3	3	3.35	3	3.41
v200_d5	4	∞	[49,92]	—	[31,52]	[26,50]	—	[25,46]	[26,35]	—	27	10.00
v200_d10	3	∞	[26,64]	—	∞	[14,29]	—	[14,19]	[14,21]	—	16	301.83
v200_d20	3	∞	[8,22]	—	10	[9,11]	—	9	9	183.40	9	184.27
v200_d30	2		[27,44]	—	∞	[6,12]	—	7	7	223.04	7	205.41
v200_d50	2		[8,15]	—	4	4	145.90	4	4	7.44	4	7.84
v200_d70	2		[4,7]	—	3	3	6.43	3	3	6.29	3	6.59
IEEE-14	5	5	5	0.01	5	5	0.01	5	5	0.01	5	0.01
IEEE-30	6	∞	14	0.01	13	13	0.01	11	11	0.01	11	0.01
IEEE-57	12	35	35	0.04	31	31	0.08	31	31	0.19	31	1.88
RTS-96	13	37	37	0.14	35	35	0.46	34	34	0.38	32	1.90
IEEE-118	14	48	48	0.15	46	46	0.25	45	45	0.25	43	1.18
IEEE-300	24	135	135	11.98	131	131	35.08	130	130	66.30	129	514.34

Notes. The case in which $s = n - 1$ is identical to the minimum strongly connected dominating set problem. We also report the objective of the dominating $(s - 2)$-club problem (Club). Here, ∞ denotes infeasibility, and blank cells indicate that $s - 2 \leq 0$.

CDS problem. This is unsurprising given that the approach taken here is very similar. However, the instance IEEE-300 takes longer here (514.34 vs. 52.88 seconds). This can be attributed to the 492.86 seconds spent in our slower callback routines.

In some applications, achieving low latency is desirable but should not be viewed as a "hard" constraint. In this case, the trade-off between CDS size and the latency guarantee should be considered. For example, the results for graphs v30_d10 and IEEE-14 show that low latency comes for free; there is a minimum CDS that also satisfies the most restrictive (but feasible) latency value s = diam. On the other hand, for the graphs v100_d30 and v150_d30, the optimal objective triples when tightening the latency parameter from $s = 3$ to $s = 2$ and may not be justified.

We also give the optimal objectives for the dominating $(s - 2)$-club problem. As observed in the introduction,

the formalization based on dominating $(s-2)$-clubs does not quite capture the intent of the latency constraints, and here we see that it usually gives the impression that no suitable low-latency CDS exists, and yet there exists a latency-s CDS. This occurs for 37 of the 47 graphs when $s = \text{diam}(G)$. An example is given in Figure 6(a). Also, Figure 6(b) shows an instance where both problems are feasible but have different optimal solutions.

5.5. Results for the Fault-Tolerant Variant

See Section 6 of the online supplement.

5.6. Results When Delays Are Node Weighted and Transmitter Based

In this section, we provide computational results for instances in which delays are node-weighted and transmitter-based. The intent is to model wireless sensor networks in which delays *depend on the transmitting node*. In our experiments, we make the simplifying assumption that the delay at node i is a given constant w_i. This is the time for node i to pass a message to any neighboring node. Following the transformation given in the introduction, this means that the edges pointing away from node i should have weight w_i. However, if one were to look at our implementation, they would see that our weights are stored by node. We prefer this representation since it is more space efficient.

To ensure that the node-based delays w_i used in our experiments are somewhat reasonable and reproducible, we define them as follows, where $\text{dist}(i,j)$ is hop-based.

$$w_i := \left\lfloor \frac{1{,}000(n-1)}{\sum_{j \in V} \text{dist}(i,j)} \right\rfloor.$$

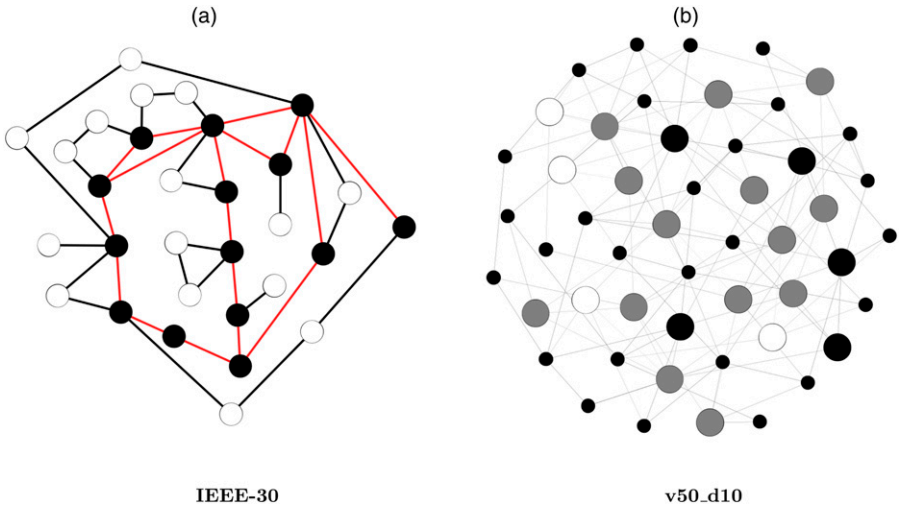
The reasoning for defining w_i in this way is as follows. So-called *central* nodes in the network will be used more frequently to transmit information, resulting in longer queueing delays. The quantity $(n-1)/\sum_{j \in V} \text{dist}(i,j)$ is the definition of (normalized) closeness centrality and the first definition for w_i that we tried. However, it is fractional, and the inexactness of later floating point calculations caused problems. To avoid exact rational arithmetic, we wanted to round closeness centrality to an integer, but this would give either a zero or a one. Multiplying by a large integer (1,000 in our case) allowed for more diverse delays.

Table 3 provides our results with these delays, where $s = \text{diam}(G)$ is set to be as restrictive as possible, while maintaining feasibility. They indicate a strength of our approach—that using weighted distances has little impact on the performance.

6. Conclusion

In this paper, we introduce a latency-constrained variant of the minimum CDS problem motivated by applications in wireless sensor networks in which one seeks a virtual backbone that provides for small end-to-end delays. We propose integer programming formulations based on length-bounded vertex cuts. These formulations generalize the best-performing existing formulations for the minimum CDS problem and generalize the best-performing formulations for the fault-tolerant variant—the minimum k-connected k-dominating set problem. A branch-and-cut implementation of formulation CUT makes easy work of synthetic instances having fewer than 100 nodes and real-life instances with up to 300 nodes, significantly outperforming formulation POLY.

Figure 6. (Color online) Examples Where the Solutions Differ Depending on Problem Formalization



Notes. Side (a) shows a minimum latency-6 CDS for IEEE-30; there is no dominating 4-club. Side (b) shows a minimum latency-5 CDS D and a minimum dominating 3-club D' for the graph v50_d10. Nodes in $D \cup D'$ are larger; nodes in $D \setminus D'$ are white; nodes in $D' \setminus D$ are black; and nodes in $D \cap D'$ are gray.

Table 3. Results for the Weighted-Distance Variant

Graph	s	BB node	H obj	Obj	Total
v30_d10	2,281	0	21	21	0.02
v30_d20	2,317	31	12	11	0.04
v30_d30	1,751	117	19	18	0.07
v30_d50	1,422	48	10	9	0.06
v30_d70	1,585	0	8	7	0.04
v50_d5	2,549	21	37	37	0.11
v50_d10	1,961	377	29	29	0.29
v50_d20	1,614	158	33	32	0.21
v50_d30	1,743	6,317	25	22	1.17
v50_d50	1,400	523	14	14	0.27
v50_d70	1,606	285	8	6	0.21
v70_d5	2,055	919	50	48	0.71
v70_d10	1,701	42	52	50	0.51
v70_d20	1,624	10,527	43	40	2.37
v70_d30	1,716	3,556	35	30	1.47
v70_d50	1,404	5,631	17	15	3.04
v70_d70	1,581	328	8	8	0.47
v100_d5	1,701	10,376	79	76	3.37
v100_d10	1,785	1,820,846	55	[47,51]	—
v100_d20	1,678	290,399	34	[21,33]	—
v100_d30	1,211	13,773	51	47	6.46
v100_d50	1,378	244,673	19	17	61.96
v100_d70	1,571	1,250	9	8	1.82
v120_d5	1,948	542,530	59	53	774.43
v120_d10	1,471	97,436	76	70	31.07
v120_d20	1,659	2,327,430	47	[37,43]	—
v120_d30	1,695	4,844,453	47	40	1,101.31
v120_d50	1,389	6,911,583	18	[14,16]	—
v120_d70	1,559	475	10	9	3.21
v150_d5	1,757	4,407,299	89	87	2,378.46
v150_d10	1,469	1,686,790	102	[81,92]	—
v150_d20	1,663	337,800	50	[30,47]	—
v150_d30	1,209	4,540,949	58	[43,50]	—
v150_d50	1,371	4,427,905	24	[18,21]	—
v150_d70	1,559	317	12	10	4.38
v200_d5	1,594	374,407	137	[80,135]	—
v200_d10	1,503	419,384	122	[83,109]	—
v200_d20	1,640	2,184,837	106	[84,96]	—
v200_d30	1,201	1,162,600	67	[45,63]	—
v200_d50	1,361	4,196,920	26	[20,23]	—
v200_d70	1,557	8,868	12	11	46.41
IEEE-14	2,154	0	8	8	0.01
IEEE-30	2,121	0	16	16	0.02
IEEE-57	2,306	0	41	41	0.11
RTS-96	2,241	285	42	41	0.51
IEEE-118	2,556	0	48	48	0.84
IEEE-300	2,646	6,558	141	137	66.90

Note. See Section 5.6 for weighting information.

Our proposed formulations are in the same vein as the recent "thin" approaches for other optimization problems (Fischetti et al. 2016, 2017). In ongoing and future research, we study the potential of using similar thin formulations based on length-bounded vertex cuts for other distance-constrained problems in networks (e.g., Salemi and Buchanan 2019).

In this paper, we focus on exact approaches. Only because our MIP solver Gurobi had problems finding feasible solutions in an hour did we employ a simple construction heuristic. Room for improvement is certainly possible, although the negative results given in Section 1 of the online supplement should not be ignored.

Acknowledgments

The authors thank three anonymous reviewers whose comments helped to significantly improve the paper's presentation.

Endnotes

[1] The complete graph is the only exception. In this case, no virtual backbone is needed, yet Definition 1 would disallow the empty set. For this reason, the complete graph is treated with generous disregard in the virtual backbone literature.

[2] An associate editor referred us to a later paper by Li and Aneja (2017) that proposes to use the same Fujie-based formulation but with some additional cuts. Li and Aneja (2017, p. 39) claim that their two implementations, named B&C1 and B&C2, "significantly outperformed the available exact algorithm[s] in the literature," but neglect to compare results with Buchanan et al. (2015). This is problematic as Li and Aneja (2017) fail to solve 2 of these 47 instances within a time limit of 3,600 seconds. Namely, Li and Aneja (2017) do not solve the instances v200_d10 and IEEE-300-Bus within the time limit, but Buchanan et al. (2015) solve these instances in 496.43 and 52.88 seconds, respectively.

[3] Schoone et al. (1987) show a related result for increasing the diameter by *edge deletions*, but it is not clear how to modify their result for our purposes. Their definition of the problem (strangely) only allows edge deletions that maintain connectivity of the graph. This allows them to perform a reduction from Hamiltonian Path by seeking subsets of $m - (n - 1)$ edges whose removal increases the diameter to $n - 1$. We feel that this is an unsatisfying hardness reduction, because a minimum cut likely has fewer edges, and its removal would make the diameter infinite. In contrast, our diameter parameter can be a small constant, and we allow for arbitrary vertex deletions.

References

Ahn N, Park S (2015) An optimization algorithm for the minimum k-connected m-dominating set problem in wireless sensor networks. *Wireless Networks* 21(3):783–792.

Alon N, Moshkovitz D, Safra S (2006) Algorithmic construction of sets for k-restrictions. *ACM Trans. Algorithms* 2(2):153–177.

Baier G, Erlebach T, Hall A, Köhler E, Kolman P, Pangrác O, Schilling H, Skutella M (2010) Length-bounded cuts and flows. *ACM Trans. Algorithms* 7(1):679–690.

Buchanan A, Sung JS, Boginski V, Butenko S (2014) On connected dominating sets of restricted diameter. *Eur. J. Oper. Res.* 236(2): 410–418.

Buchanan A, Sung JS, Butenko S, Pasiliao EL (2015) An integer programming approach for fault-tolerant connected dominating sets. *INFORMS J. Comput.* 27(1):178–188.

Chen S, Ljubić I, Raghavan S (2010) The regenerator location problem. *Networks* 55(3):205–220.

Chen S, Ljubić I, Raghavan S (2015) The generalized regenerator location problem. *INFORMS J. Comput.* 27(2):204–220.

Dinur I, Steurer D (2014) Analytical approach to parallel repetition. Shmoys D, ed. *Proc. 46th Annual ACM Sympos. Theory Comput.* (ACM, New York), 624–633.

Du D-Z, Wan P-J (2013) *Connected Dominating Set: Theory and Applications*, vol. 77 (Springer, New York).

Edmonds J, Johnson EL (1970) Matching: a well-solved class of integer linear programs. Guy R, Hanani H, Sauer N, Schönheim, eds. *Proc. Calgary Internat. Conf. Combin. Structures Their Appl* (Gordon and Breach, Philadelphia), 89–92.

Fan N, Watson J-P (2012) Solving the connected dominating set problem and power dominating set problem by integer programming. Lin G, ed. *Combinatorial Optimization and Applications*, vol. 7402 (Springer, New York), 371–383.

Feige U (1998) A threshold of ln n for approximating set cover. *J. ACM* 45(4):634–652.

Fischetti M, Leitner M, Ljubić I, Luipersbeck M, Monaci M, Resch M, Salvagnin D, Sinnl M (2017) Thinning out Steiner trees: a node-based model for uniform edge costs. *Math. Programming Comput.* 9(2):203–229.

Fischetti M, Ljubić I, Sinnl M (2016) Redesigning Benders decomposition for large-scale facility location. *Management Sci.* 63(7):2146–2162.

Fujie T (2004) The maximum-leaf spanning tree problem: formulations and facets. *Networks* 43(4):212–223.

Garey MR, Johnson DS (1979) *Computers and Intractability: A Guide to the Theory of NP-Completeness* (WH Freeman and Company, New York).

Gendron B, Lucena A, da Cunha AS, Simonetti L (2014) Benders decomposition, branch-and-cut, and hybrid algorithms for the minimum connected dominating set problem. *INFORMS J. Comput.* 26(4):645–657.

Grötschel M, Lovász L, Schrijver A (1993) *Geometric Algorithms and Combinatorial Optimization*, Algorithms and Combinatorics, vol. 2 (Springer, Berlin).

Haynes TW, Hedetniemi S, Slater P (1998) *Fundamentals of Domination in Graphs* (CRC Press, Clearwater, FL).

Impagliazzo R, Paturi R (2001) On the complexity of k-SAT. *J. Comput. System Sci.* 62:367–375.

Impagliazzo R, Paturi R, Zane F (2001) Which problems have strongly exponential complexity? *J. Comput. System Sci.* 63:512–530.

Krishnamoorthy MS, Krishnamurthy B (1987) Fault diameter of interconnection networks. *Comput. Math. Appl.* 13(5):577–582.

Li D, Du H, Wan PJ, Gao X, Zhang Z, Wu W (2009) Construction of strongly connected dominating sets in asymmetric multihop wireless networks. *Theoret. Comput. Sci.* 410(8–10):661–669.

Li X, Aneja YP (2017) Regenerator location problem: Polyhedral study and effective branch-and-cut algorithms. *Eur. J. Oper. Res.* 257(1): 25–40.

Li Y, Kim D, Zou F, Du D-Z (2008) Constructing minimum connected dominating sets with bounded diameters in wireless networks. *IEEE Trans. Parallel Distributed Systems* 20(2):147–157.

Lovász L, Neumann-Lara V, Plummer M (1978) Mengerian theorems for paths of bounded length. *Periodica Math. Hungarica* 9(4): 269–276.

Lucena A, Maculan N, Simonetti L (2010) Reformulations and solution algorithms for the maximum leaf spanning tree problem. *Comput. Management Sci.* 7(3):289–311.

Lund C, Yannakakis M (1994) On the hardness of approximating minimization problems. *J. ACM* 41(5):960–981.

Martin RK (1991) Using separation algorithms to generate mixed integer model reformulations. *Oper. Res. Lett.* 10(3):119–128.

Morgan MJ, Grout V (2008) Finding optimal solutions to backbone minimisation problems using mixed integer programming. *Proc. 7th Internat. Network Conf.* (Glyndŵr University Research Online), 53–64.

Moshkovitz D (2015) The projection games conjecture and the NP-hardness of ln n-approximating set-cover. *Theory Comput.* 11(7): 221–235.

Orlin JB (2013) Max flows in $O(nm)$ time, or better. Boneh D, Roughgarden T, Feigenbaum J, eds. *Proc. 45th Annual ACM Sympos. Theory Comput.* (ACM, New York), 765–774.

Raz R, Safra S (1997) A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP. Raz R, Safra S, eds. *Proc. 25th Annual ACM Sympos. Theory Comput.* (ACM, New York), 475–484.

Salemi H, Buchanan A (2019) Parsimonious formulations for low-diameter clusters. Accessed October 3, 2019, http://www.optimization-online.org/DB_HTML/2017/09/6196.html.

Sassano A (1989) On the facial structure of the set covering polytope. *Math. Program.* 44(1–3):181–202.

Schaudt O (2013) On dominating sets whose induced subgraphs have a bounded diameter. *Discrete Appl. Math.* 161(16):2647–2652.

Schoone AA, Bodlaender HL, Van Leeuwen J (1987) Diameter increase caused by edge deletion. *J. Graph Theory* 11(3): 409–427.

Schrijver A (2003) Combinatorial optimization: polyhedra and efficiency. *Algorithms and Combinatorics*, vol. 24 (Springer, Berlin).

Simonetti L, Da Cunha AS, Lucena A (2011) The minimum connected dominating set problem: formulation, valid inequalities and a branch-and-cut algorithm. Pahl J, Reiners T, Voß S, eds. *Internat. Conf. Network Optim.* (Springer, Berlin), 162–169.

Thorup M (2004) Integer priority queues with decrease key in constant time and the single source shortest paths problem. *J. Comput. System Sci.* 69(3):330–353.

Veremyev A, Boginski V (2012) Identifying large robust network clusters via new compact formulations of maximum k-club problems. *Eur. J. Oper. Res.* 218(2):316–326.

Xie M, Haenggi M (2009) Toward an end-to-end delay analysis of wireless multihop networks. *Ad Hoc Networks* 7(5):849–861.

Xu J (2001) *Topological Structure and Analysis of Interconnection Networks* (Kluwer, Amsterdam).

Xu M, Xu JM, Hou XM (2005) Fault diameter of Cartesian product graphs. *Inform. Processing Lett.* 93(5):245–248.

Zhang N, Shin I, Zou F, Wu W, Thai MT (2008) Trade-off scheme for fault tolerant connected dominating sets on size and diameter. *Proc. 1st ACM Internat. Workshop Foundations Wireless Ad Hoc Sensor Networking Comput.* (ACM, New York), 1–8.

Zhong Y, Haenggi M, Zheng FC, Zhang W, Quek TQ, Nie W (2017) Toward a tractable delay analysis in ultradense networks. *IEEE Comm. Magazine* 55(12):103–109.