An Integer Programming Approach for Fault-Tolerant Connected Dominating Sets

Article *in* Informs Journal on Computing · February 2015

DOI: 10.1287/ijoc.2014.0619

Austin L Buchanan
Oklahoma State University - Stillwater

Je Sang Sung
Texas A&M University

Sergiy Butenko
Texas A&M University

Eduardo Pasiliao
Air Force Research Laboratory

An Integer Programming Approach for Fault-Tolerant Connected Dominating Sets*

Austin Buchanan Je Sang Sung Sergiy Butenko
Eduardo L. Pasiliao

March 2, 2016

Abstract

This paper considers the minimum k-connected d-dominating set problem, which is a fault-tolerant generalization of the minimum connected dominating set (MCDS) problem. Three integer programming formulations based on vertex-cuts are proposed (depending on whether $d < k$, $d = k$, or $d > k$) and their integer hulls are studied. The separation problem for the vertex-cut inequalities is a weighted vertex-connectivity problem and is polytime solvable, meaning that the LP relaxation can be solved in polytime despite having exponentially many constraints. A new class of valid inequalities – r-robust vertex-cut inequalities – is introduced and is shown to induce exponentially many facets. Finally, a lazy-constraint approach is shown to compare favorably with existing approaches for the MCDS problem (the case $k = d = 1$), and is in fact the fastest in literature for standard test instances. A key subroutine is an algorithm for finding an inclusion-wise minimal vertex-cut in linear time. Computational results for $(k, d) = (2, 1), (2, 2), (3, 3), (4, 4)$ are provided as well.

1 Introduction

In the context of wireless ad-hoc networks, a connected dominating set (CDS) is often created to serve as a virtual backbone for the network (Du and Wan 2013). The vertices in the network may not be able to communicate with each other directly; however, a message departing from one vertex can be transmitted through intermediate vertices to reach its destination. These intermediate vertices are the CDS.

The properties of a CDS ensure the minimum requirements for a functioning virtual backbone. Some other potentially desirable characteristics include low

*Published as A. Buchanan, J.S. Sung, S. Butenko, E.L. Pasiliao. An integer programming approach for fault-tolerant connected dominating sets. *INFORMS Journal on Computing*, 27(1):178-188, 2015. http://dx.doi.org/10.1287/ijoc.2014.0619

latency (the message transmission time is short) and fault-tolerance (the failure of some vertices does not disrupt service). The latency issue can be remedied by requiring that the subgraph induced by the CDS has a small diameter. This has been explored by Buchanan et al. (2014). In this paper, we consider the issue of fault-tolerance, or robustness under vertex failure, by studying k-connected d-dominating sets. Note that a k-connected k-dominating set remains a CDS if any fewer than k vertices fail.

Usually one is interested in a small CDS, leading to the minimum CDS (MCDS) problem. The NP-hard MCDS problem and the closely-related maximum leaf spanning tree problem are well-studied in the operations research and computer science literature, see, e.g., a recent book (Du and Wan 2013) on the topic. A variety of approaches have been considered, including exact approaches (Fomin et al. 2008, Lucena et al. 2010, Simonetti et al. 2011, Morgan and Grout 2008, Fan and Watson 2012, Fujie 2004, 2003, Chen et al. 2010, Gendron et al. 2014), heuristics (Butenko et al. 2004, Blum et al. 2005), approximation algorithms (Guha and Khuller 1998, Lu and Ravi 1998), and polynomial-time approximation schemes for unit-disk graphs (Cheng et al. 2003, Hunt et al. 1998) and for unit-ball graphs (Zhang et al. 2008).

For the minimum k-connected d-dominating set problem, there exist approximation algorithms with a constant factor for unit disk graphs and approximation algorithms with a logarithmic factor for some values of d and k for arbitrary graphs (Dai and Wu 2006, Shang et al. 2007, Thai et al. 2007, Wu and Li 2008, Li et al. 2012, Zhou et al. 2014). do Forte et al. (2013) provide IP formulations and take a branch-and-cut approach for the minimum 2-1-CDS problem. Ahn and Park (2014) study the minimum k-d-CDS problem and provide an approach to solve the problem that is similar to the lazy-constraint approach taken in this paper.

Given a graph $G = (V, E)$, a subset $S \subseteq V$ of vertices is called a k-connected d-dominating set (k-d-CDS) if the subgraph induced by S has vertex-connectivity at least k, and every vertex outside of S has at least d neighbors from S. The minimum k-connected d-dominating set problem is the focus of this paper. This problem asks: given a graph and positive integers d and k, either find a k-d-CDS of the smallest size in the graph, or determine that none exist. It is easy to see that k-d-CDS generalizes CDS (set $k = d = 1$), and the minimum k-d-CDS problem generalizes the classical MCDS problem.

Many results in this paper concerning k-d-CDS and the k-d-CDS polytope are stated for $d \geq k$. This ensures nice properties that no longer hold when $d < k$. In particular, it is shown that every superset of a k-d-CDS is also a k-d-CDS when $d \geq k$; however, this can fail when $d < k$.

1.1 Notation and terminology

We consider a simple undirected graph $G = (V, E)$ with set V of n vertices and set E of m edges. We denote the (open) neighborhood of a vertex $i \in V$ by $N(i) = \{j \in V \mid \{i, j\} \in E\}$, and the closed neighborhood of $i \in V$ by $N[i] = N(i) \cup \{i\}$. A vertex $v \in V$ is said to be universal if $N[v] = V$. A set

$D \subseteq V$ is called a dominating set if each vertex in $V \setminus D$ has a neighbor in D. Let $G[S]$ denote the subgraph induced by $S \subseteq V$. A dominating set that induces a connected graph is called a connected dominating set (CDS). A graph is said to be k-vertex-connected (or, simply, k-connected) if there exist at least k vertex-disjoint paths between every pair of distinct vertices. Equivalently, a connected graph is k-connected if it remains connected and nontrivial after the removal of fewer than k vertices. The vertex-connectivity $\kappa(G)$ of a graph G is defined as the largest integer k such that G is k-connected. By convention, a single universal vertex does not constitute a CDS since its induced subgraph is trivial. However, if one considers a single vertex to constitute a CDS, its existence can easily be determined. Similarly, a subset of k universal vertices does not constitute a k-k-CDS since its induced subgraph becomes trivial after the removal of $k-1$ vertices; but, if desired, one can determine whether such a set exists in linear time.

Definition 1. *Given a graph $G = (V, E)$, a subset $S \subseteq V$ of vertices is called a k-connected d-dominating set (k-d-CDS) if $\kappa(G[S]) \geq k$ and $|N(i) \cap S| \geq d$ $\forall i \in V \setminus S$. The minimum k-d-CDS problem is to find a smallest subset $S \subseteq V$ of vertices such that S is a k-connected d-dominating set, or decide that none exist.*

A vertex-cut $C \subseteq V$ of a connected graph G is a subset of vertices such that $G[V \setminus C]$ has at least two connected components or is trivial. A vertex-cut C is said to be minimal (by inclusion) if no proper subset of C is a vertex-cut. For distinct vertices $a, b \in V$, an a-b separator is a subset $S \subseteq V \setminus \{a, b\}$ of vertices, such that a and b are disconnected in $G[V \setminus S]$.

1.2 Existing polyhedral results for 1-1-CDS

Several formulations have been proposed for MCDS and for the maximum leaf spanning tree problem (MLSTP). Many of the formulations (called Edge-Vertex Formulations) have a binary variable for each edge and for each vertex of the graph (Fujie 2003, 2004, Lucena et al. 2010, Simonetti et al. 2011). In constrast, formulations (called Vertex Formulations) found in (Fujie 2004, Yuan 2005, Ahn and Park 2014) only have binary variables associated with the vertices. Polyhedral studies have been conducted for both types of formulations for the maximum leaf spanning tree problem (Fujie 2004). We are particularly interested in the Vertex Formulations, as the formulations proposed in this paper have this form. It has been noted that the maximum leaf spanning tree polytope is full-dimensional if and only if the graph is biconnected (Fujie 2004). Moreover, this study identified conditions for when the simple 0-1 bounds induce facets. A characterization of facet-defining vertex-cut inequalities was also established.

The study presented in this paper generalizes these results (translated to the CDS context) for k-d-CDS when $d \geq k$. In addition, a new class of valid inequalities – called r-robust vertex-cut inequalities – is introduced. These inequalities generalize inequalities for MCDS introduced by Gendron et al. (2014), which are in turn a generalization of the vertex-cut inequalities. We show that this generalization leads to a simple characterization for when the vertex-cut inequalities induce facets of the 1-1-CDS polytope. Next, we provide a brief summary of results presented in each section.

1.3 Our contributions

In Section 2, we explore some properties of k-d-CDS, including the cost of ensuring robustness and show that a graph has a k-d-CDS (for $d \geq k$) if and only if it is k-connected. More generally, a k-d-CDS exists if and only if the graph has a k-block that is d-dominating—and this can be determined in polynomial time. A characterization of k-d-CDS proved in this section leads directly to the IP formulations. In Section 3, three exponentially-sized formulations for the minimum k-d-CDS problem are presented that are based on vertex-cuts, and their integer hulls are studied. Even though the three formulations have exponentially many constraints, their LP relaxations can be solved in polynomial time since the separation problems are weighted vertex-cut problems. A generalization of the vertex-cut inequality, called an r-robust vertex-cut inequality, is introduced, and the class of all such inequalities is shown to induce exponentially many facets. In Section 4, a lazy-constraint approach is described and computational results are provided for the minimum k-d-CDS problem for several values of k and d. A key subroutine finds an (inclusion-wise) minimal vertex-cut in linear time. The computational results for the classical MCDS problem ($k = d = 1$) are the fastest in literature. In fact, the lazy-constraint approach solves 42 of the 47 problem instances in a 10-second time limit, while no previous approach solves this many in a one-hour time limit.

2 Properties of k-d-CDS

Lemma 1. *Let d and k be positive integers such that $d \geq k$. For any vertex-cut $C \subset V$ and k-d-CDS $S \subseteq V$ of a graph $G = (V, E)$, we have that $|S \cap C| \geq k$.*

Proof. Proof. For contradiction purposes, suppose that there exists a k-d-CDS $S \subseteq V$ of G with $|S \cap C| \leq k-1$. Since C is a vertex-cut, it either separates the graph into at least two connected components or it is a collection of $n-1$ or n vertices. It is clear that if $|C| = n-1$ or $|C| = n$, then S cannot be a k-d-CDS since $|S| = |S \cap C| + |S \cap (V \setminus C)| \leq (k-1)+1 = k$ and a k-d-CDS must have at least $k+1$ vertices to be k-connected. So we can assume that $C \subseteq V$ separates $A, B \subseteq V \setminus C$, meaning that $A \cap B = \emptyset$ and $E \cap \{\{u, v\} \mid u \in A, v \in B\} = \emptyset$. We consider two cases. In the first case, suppose that $A \cap S = \emptyset$ (or $B \cap S = \emptyset$). Then S is not a k-d-CDS, since no vertex belonging to A (B) will be k-dominated (hence, not d-dominated). In the second case, suppose that $A \cap S \neq \emptyset$ and $B \cap S \neq \emptyset$, then there exists $a \in A \cap S$ and $b \in B \cap S$. Then the removal of the at most $k-1$ nodes in $S \cap C$ disconnects a and b in $G[S]$, i.e., $G[S]$ is not k-vertex-connected. Thus S cannot be a k-d-CDS, a contradiction. □

Theorem 1 (Characterization of k-d-CDS for $d \geq k$)**.** *Let d and k be positive integers such that $d \geq k$. Given a graph $G = (V, E)$, a subset $S \subseteq V$ of vertices is a k-d-CDS of G if and only if*

1. *$|S \cap C| \geq k$ for every vertex-cut $C \subset V$ of G, and*
2. *$|S \cap N(v)| \geq d$ for every vertex $v \in V \setminus S$.*

Moreover, $S \subseteq V$ is a k-k-CDS for G if and only if the first condition above holds.

Proof. Proof. The 'only if' direction follows by Lemma 1 and by definition of k-d-CDS. For the other direction, suppose that $|S \cap C| \geq k$ for every vertex-cut $C \subset V$ of G. We will show that $|C'| \geq k$ for every vertex-cut $C' \subseteq S$ of $G[S]$. See that $C' \cup (V \setminus S)$ is a vertex-cut for G. Accordingly, by assumption, we have that $|S \cap (C' \cup (V \setminus S))| \geq k$. Hence, $|C'| = |S \cap C'| = |S \cap (C' \cup (V \setminus S))| \geq k$ and $G[S]$ is k-connected. If $|S \cap N(v)| \geq d$ for every vertex $v \in V \setminus S$, then S is also d-dominating; hence it is a k-d-CDS for G. See that if $k = d$, then 1 implies 2. This holds because, for each vertex $v \in V$, its neighborhood $N(v)$ is a vertex-cut, so v will be $d(= k)$-dominated. $\square$

Proposition 1. *For every $d \geq k$, the collection of k-d-CDSs of a k-connected graph is nonempty and closed under taking supersets. For every $d < k$, there exists a graph for which the collection of k-d-CDSs is not closed under taking supersets.*

Proof. Proof. For $d \geq k$, the set V will be a k-d-CDS of $G = (V, E)$ when G is k-connected, so the collection is nonempty. Now, given a k-d-CDS $S \subset V$ and $S' \supset S$, we have that: $|S' \cap C| \geq |S \cap C| \geq k$ for every vertex-cut $C \subset V$; and $|S' \cap N(v)| \geq |S \cap N(v)| \geq d$ for every vertex $v \in V \setminus S' \subset V \setminus S$. Thus, by Theorem 1, S' is also a k-d-CDS, and the collection of k-d-CDSs is closed under taking supersets.

Now, consider $d < k$ and a complete graph on vertices $S = \{1, 2, \ldots, k+1\}$ with an additional vertex v attached to the first d vertices. While S forms a k-d-CDS, $S \cup \{v\}$ does not. $\square$

It is easy to see that a CDS exists if and only if the graph is connected. This can be generalized for k-d-CDS as follows. Recall that a k-block is an inclusion-wise maximal k-connected subset of vertices. Every n-vertex graph has at most $\lfloor (2n-1)/3 \rfloor$ k-blocks, and they can be listed in polynomial time (Matula 1978). As a result, for all positive integers k and d, there is a polynomial time algorithm to find a k-d-CDS (when one exists).

Corollary 1 (Characterization of graphs with a k-d-CDS)**.** *A graph $G = (V, E)$ has a k-d-CDS if and only if it has a k-block that is d-dominating. Further, for $d \geq k$, the following are equivalent.*

1. *There exists a k-d-CDS in G.*

2. *The vertex set V is a k-d-CDS for G.*

3. *The graph G is k-connected.*

Proof. Proof. Suppose G has a k-d-CDS $S \subseteq V$. Then S is a subset of a maximal k-block S'. Note that S' will also be d-dominating, so S' is a k-d-CDS as well. The direction 'if' is trivial. That the other statements are equivalent follows by Theorem 1 and Proposition 1. □

2.1 The cost of fault-tolerance

Fault-tolerance, or robustness under vertex failure, is a desirable property; however, it is not free. In fact, as Corollary 2 shows, fault-tolerance entails a strictly more-costly solution. One may wonder if the extra cost required for fault-tolerance can be bounded. Proposition 2 shows that this is not the case; the cost can be essentially as large as possible. First, we make use of the following lemma, which holds trivially by definition.

Lemma 2. *Let k be a positive integer. Suppose that a graph $G = (V, E)$ has a $(k+1)$-$(k+1)$-CDS $S \subseteq V$. Then for any $v \in S$, we have that $S \setminus \{v\}$ is a k-k-CDS for G.*

Corollary 2. *Let k be a positive integer. Suppose that a graph $G = (V, E)$ has a $(k+1)$-$(k+1)$-CDS. Denote by $\gamma_{k,d}(G)$ the size of a minimum k-d-CDS of G and by $\gamma_c(G)$ the size of a minimum CDS in G. Then,*

$$k + \gamma_c(G) \leq 1 + \gamma_{k,k}(G) \leq \gamma_{k+1,k+1}(G). \tag{1}$$

Moreover, each inequality is sharp on a complete graph on $k+2$ vertices.

Proof. Proof. To see the rightmost inequality, let S be a minimum $(k+1)$-$(k+1)$-CDS and apply Lemma 2. The first inequality holds by induction on k since $\gamma_c(G) = \gamma_{1,1}(G)$. Sharpness holds for K_{k+2} because $\gamma_{p,p}(K_{k+2}) = p+1$ for any $1 \leq p \leq k+1$. □

Proposition 2. *For any fixed positive integer k, there exist infinitely many graphs $G = (V, E)$ for which $\gamma_{k,k}(G)$ is a constant, but $\gamma_{k+1,k+1}(G) = |V| - 2$.*

Proof. Proof. For any $q \geq 4$ and any $k \geq 1$, we construct a graph G on $k+q$ vertices such that $\gamma_{k,k}(G) = k+1$ and $\gamma_{k+1,k+1}(G) = k+q-2$. Then, to prove the claim, construct such a graph for each integer $q \geq 4$. The graph $G = (Q \cup R, E \cup F)$ is constructed as follows:

$$\begin{aligned}
Q &= \{u_1, \ldots, u_q\} \\
R &= \{v_1, \ldots, v_k\} \\
E &= \{\{v, x\} : v \neq x, v \in R, x \in Q \cup R\} \\
F &= \{\{u_i, u_{i+1}\} : 1 \leq i < q\}.
\end{aligned}$$

It is clear that $\gamma_{k,k}(G) = k+1$, since $R \cup \{u_1\}$ is a k-k-CDS for G. Also see that $\gamma_{k+1,k+1}(G) \leq k+q-2$ since $R \cup \{u_i : 1 < i < q\}$ is a $(k+1)$-$(k+1)$-CDS. Now we show that $\gamma_{k+1,k+1}(G) \geq k+q-2$. It is clear that $R \cup \{u_i\}$ is a u_1-u_q separator of size $k+1$ for any $1 < i < q$. Thus, by Lemma 1, every $(k+1)$-$(k+1)$-CDS for G must use every vertex from $R \cup \{u_i\}$ for $1 < i < q$. □

3 IP Formulations and the k-d-CDS polytope

In this section, we describe and study three different formulations for the minimum k-d-CDS problem. The formulations differ depending on whether $d = k$; $d > k$; or $d < k$ and are similar to those proposed by Ahn and Park (2014). In each formulation, the n-vector x of 0-1 variables is the characteristic vector of a k-d-CDS for a graph $G = (V, E)$.

In the case that $d = k$, the formulation has the following representation.

$$\gamma_{k,k}(G) = \min \sum_{v \in V} x_v \tag{2}$$

$$\sum_{v \in C} x_v \geq k \text{ for every minimal vertex-cut } C \subset V \tag{3}$$

$$x_v \in \{0,1\} \ \forall v \in V \tag{4}$$

This formulation generalizes the formulation for MCDS given by Yuan (2005), which is itself obtained by replacing each variable y_i in the maximum leaf spanning tree formulation of Fujie (2004) with $1 - x_i$. The validity of the formulation follows by Theorem 1. Recognize that the formulation is still valid if constraint (3) is required for every vertex-cut; however, the minimal vertex-cut inequalities subsume the non-minimal vertex-cut inequalities. We note that there can be exponentially many *minimal* vertex-cut constraints, since there exist k-connected graphs with $\Omega\left(2^k \frac{n^2}{k^2}\right)$ *minimum* vertex-cuts (Kanevsky 1990).

When $d > k$, we add one more set of constraints to ensure d-domination:

$$\gamma_{k,d}(G) = \min \sum_{v \in V} x_v \tag{5}$$

$$\sum_{v \in C} x_v \geq k \text{ for every minimal vertex-cut } C \subset V \tag{6}$$

$$(d-k)x_v + \sum_{j \in N(v)} x_j \geq d \ \forall v \in V \tag{7}$$

$$x_v \in \{0,1\} \ \forall v \in V \tag{8}$$

It would have been sufficient to add just the d-domination constraints $dx_v + \sum_{j \in N(v)} x_j \geq d$. The validity of such a formulation would follow by Theorem 1. However, the inequality $(d-k)x_v + \sum_{j \in N(v)} x_j \geq d$ is tighter. This tighter inequality is valid since, if $x_v = 0$, then the constraint is the same, and if $x_v = 1$, then at least k neighbors of v must be selected for k-connectivity. This tightening generalizes a previous observation of Buchanan et al. (2014) for CDS and has also been noted for k-d-CDS by Ahn and Park (2014).

When $d < k$, the vertex-cut constraint that was used in the two previous formulations is no longer valid. However, a simple modification to the vertex-cut constraint makes it valid again. If desired, the quadratic vertex-cut constraint (10) can be linearized.

When $d < k$:

$$\gamma_{k,d}(G) = \min \sum_{v \in V} x_v \tag{9}$$

$$\sum_{v \in S} x_v \geq k x_a x_b \text{ for every minimal } a\text{-}b\text{-separator } S \subset V,\ \forall a, b \in V,\ a \neq b, \{a, b\} \notin E \tag{10}$$

$$(d - k)x_v + \sum_{j \in N(v)} x_j \geq d \ \forall v \in V \tag{11}$$

$$x_v \in \{0, 1\} \ \forall v \in V \tag{12}$$

It is known that the CDS polytope is full-dimensional if and only if the graph is biconnected (Fujie 2004). This can be generalized for the k-d-CDS polytope when $d \geq k$. The k-d-CDS polytope P_I for a graph G is defined as the convex hull of characteristic vectors of k-d-CDSs of G.

Theorem 2 (Integer hull dimension, 0-1 facets)**.** *Let $d \geq k$ be positive integers. The k-d-CDS polytope for a graph $G = (V, E)$ is full-dimensional if and only if G is $(k + 1)$-connected with minimum degree at least d. Moreover, if G is $(k + 1)$-connected with minimum degree at least d, then for every vertex $v \in V$,*

1. *$x_v \leq 1$ induces a facet, and*
2. *$x_v \geq 0$ induces a facet if and only if*
   - *i. v does not belong to a vertex-cut of size $k + 1$, and*
   - *ii. $|N(w)| \geq d + 1$ for every vertex $w \in N[v]$.*

More generally, the k-d-CDS polytope of a k-connected n-vertex graph $G = (V, E)$ has dimension $n - |\Psi_{k,d}(G)|$, where

$$\Psi_{k,d}(G) = \{v \in V \mid v \text{ belongs to a vertex-cut of size } k \text{ or } |N(v)| < d\}.$$

Proof. Proof. The proof is straightforward (but tedious), so we provide a short sketch. The key fact is that, when $d \geq k$, the collection of k-d-CDSs of a graph is closed under taking supersets (by Proposition 1). To prove dimension, consider the usual $n - |\Psi_{k,d}(G)| + 1$ affinely independent points in the k-d-CDS polytope P_I. The points include the all-ones vector and the all-ones vectors with one entry flipped to zero, i.e., $\mathbf{1}$ and $\mathbf{1} - e_v,\ v \in V \setminus \Psi_{k,d}(G)$, where e_v is the zero vector with v^{th} entry changed to one. Note that a vertex $v \in \Psi_{k,d}(G)$ must belong to every k-d-CDS of G (forcing $x_v = 1$) and thus the dimension is reduced by $|\Psi_{k,d}(G)|$. So, the dimension is at least, and at most, $n - |\Psi_{k,d}(G)|$. Finally, it is easy to see that the set $\Psi_{k,d}(G)$ is empty (i.e., P_I is full-dimensional) if and only if the graph is $(k+1)$-connected with minimum degree at least d. When P_I is full-dimensional, the points given earlier show point 1. Proving point 2 reduces to checking that $\mathbf{1} - e_v - e_z$ is feasible for each $z \in V \setminus \{v\}$. □

Proposition 3. *The separation problem for the vertex-cut inequalities* (3) *and* (6) *is a weighted vertex-connectivity problem. It can be solved in* $O(\kappa nm \log(n^2/m))$ *time (Henzinger et al. 2000), where* $\kappa \leq m/n$ *is the unweighted vertex connectivity.*

Proof. Proof. Given $x' \in [0,1]^n$ and a graph $G = (V, E)$, the separation problem asks if all vertex-cut inequalities are satisfied by x', and, if not, requires a vertex-cut $C \subset V$ of G with $\sum_{i \in C} x'_i < k$. Construct an instance of weighted vertex connectivity by giving each vertex $i \in V$ weight x'_i. It is clear that the weighted vertex-connectivity of G is at least k if and only if each vertex-cut inequality (3) or (6) is satisfied. If the weighted vertex-connectivity of G is less than k, a vertex-cut C having weight less than k will be found and the corresponding inequality $\sum_{i \in C} x_i \geq k$ is valid and separates x' from the k-d-CDS integer hull. □

Thus, when $d \geq k$ the k-d-CDS linear programming relaxation can be solved in polynomial time, e.g., using the ellipsoid method (Khachian 1979, Grötschel et al. 1981). We note that the vertex-cut inequality (10) for $d < k$ can be linearized to $\sum_{v \in C} x_v \geq k(x_a + x_b - 1)$. The separation problem for these linearized inequalities can also be solved in polynomial time. In a naive approach, solve $\binom{n}{2}$ minimum weight a-b-vertex-cut problems by reducing them to edge-connectivity problems and use a max-flow algorithm to solve the transformed problem. Regardless, this does not seem useful since the linearized inequalities will be very weak.

Unfortunately, the minimal vertex-cut inequalities do not necessarily induce facets. As an example, consider the 1-1-CDS polytope for C_6 (a cycle on 6 vertices) with vertices $\{1,2,3,4,5,6\}$ and edge set

$$\{\{1,2\},\{2,3\},\{3,4\},\{4,5\},\{5,6\},\{6,1\}\}.$$

The minimal vertex-cut inequality $x_1 + x_3 \geq 1$ is dominated by the facet-defining inequality $x_1 + x_3 + x_5 \geq 2$. The possible failure of the minimal vertex-cut inequalities to induce facets motivates the study of r-robust vertex-cuts. We note that a vertex-cut is equivalent to a 0-robust vertex-cut. The facet-defining inequality $x_1 + x_3 + x_5 \geq 2$ described earlier is actually a 1-robust vertex-cut inequality.

Definition 2 (r-robust vertex-cut)**.** *Let r be a nonnegative integer. A subset $C \subset V$ of vertices is said to be an r-robust vertex-cut if $C \setminus C'$ is a vertex-cut for any $C' \subset C$ with $|C'| \leq r$.*

The following set of valid inequalities generalizes previously known inequalities for CDS (see Proposition 3 of Gendron et al. (2014)).

Proposition 4 (r-robust vertex-cut inequalities)**.** *Let $d \geq k$ be positive integers. The following inequality is valid for k-d-CDS for an r-robust vertex-cut $C \subseteq V$.*

$$\sum_{i \in C} x_i \geq k + r \tag{13}$$

Proof. Proof. The proof is by induction on r. The base case where $r = 0$ is exactly the vertex-cut inequality. For the inductive step, suppose that the r-robust vertex-cut inequalities hold. We shall show that the $(r+1)$-robust vertex-cut inequalities are valid. Consider an arbitrary $(r+1)$-robust vertex-cut C. It follows that $C \backslash \{v\}$ is an r-robust vertex-cut for any $v \in C$. Accordingly, we have that $\sum_{i \in C \backslash \{v\}} x_i \geq k+r$ for any vertex $v \in C$. Summing these inequalities yields

$$\sum_{i \in C} (|C| - 1) x_i \geq |C|(k + r). \tag{14}$$

Then, dividing by $|C| - 1$, and by the integrality of the left-hand-side we have

$$\sum_{i \in C} x_i \geq \left\lceil \frac{|C|}{|C| - 1}(k + r) \right\rceil = \left\lceil \frac{|C| - 1}{|C| - 1}(k + r) + \frac{1}{|C| - 1}(k + r) \right\rceil = k + r + 1. \tag{15}$$

The last equality holds because $k + r$ is positive, yet $k + r \leq |C| - 1$. □

Now that the r-robust vertex-cut inequalities are shown to be valid, we can establish that they are actually interesting. Namely, the class of all such inequalities can induce exponentially many facets. It is shown that for every maximal independent set of a cycle, a facet-defining r-robust vertex-cut inequality can be generated. The number of maximal independent sets of a cycle grows exponentially in the number of vertices.

Lemma 3 (Füredi (1987))**.** *The number of maximal independent sets of the n-vertex cycle graph is $P(n)$, where $P(3) = 3$, $P(4) = 2$, $P(5) = 5$, and $P(n) = P(n-2) + P(n-3)$ for $n \geq 6$. These numbers are, in fact, the Perrin numbers, which grow at the rate $P(n) \sim \rho^n$, where $\rho = \lim_{n \to \infty} \frac{P(n)}{P(n-1)} = 1.3247\ldots$ is the plastic number (see, e.g., Weisstein (2013)).*

Proposition 5. *The number of facet-defining r-robust vertex-cut inequalities (summed over all values of r) of the 1-1-CDS polytope of the n-vertex cycle graphs grows exponentially in n.*

Proof. Proof. We demonstrate that every independent set S of C_n ($n \geq 3$) is an $(|S| - 2)$-robust vertex-cut. Moreover, the associated $(|S| - 2)$-robust vertex-cut inequality for a *maximal* independent set S is shown to induce a facet of the 1-1-CDS polytope of C_n. Thus, by Lemma 3, the number of such inequalities grows exponentially in the class of cycles.

Claim 1: an independent set S of C_n is an $(|S|-2)$-robust vertex-cut. It is clear that any pair of nonadjacent vertices forms a vertex-cut for C_n. Thus, for any $F \subset S$ with $|F| \leq |S|-2$, we have that $S \setminus F$ is a vertex-cut for C_n, implying that S is indeed an $(|S|-2)$-robust vertex-cut.

Claim 2: the $(|S|-2)$-robust vertex-cut inequality for a maximal independent set S of C_n is facet-defining for the 1-1-CDS polytope. The 1-1-CDS polytope of C_n ($n \geq 3$) is full-dimensional since C_n is biconnected. We provide n affinely independent points. See that the point $\mathbf{1} - e_i$ is feasible for each $i \in S$. Now we generate $n - |S|$ other points. Note that each vertex $v \notin S$ has at least one neighbor in S, since otherwise $S \cup \{v\}$ would be independent, contradicting the maximality assumption. So for each $v \notin S$ there exists an adjacent vertex $s_v \in S$, and the point $\mathbf{1} - e_v - e_{s_v}$ is feasible. It is not hard to see that these $n-|S|$ points along with the $|S|$ points from earlier are affinely independent. $\square$

Now that r-robust vertex-cut inequalities have been shown to be interesting (at least in the class of cycles), we would like to know when they will be facet-defining in arbitrary graphs. We first mention a theorem of Fujie (2004) showing when a particular inequality defines a facet of the maximum leaf spanning tree polytope. The binary variable y_i takes a value of one if and only if vertex i is a leaf of the spanning tree (i.e., does not belong to the CDS).

Theorem 3 (Theorem 3.9 of Fujie (2004))**.** *Let $G = (V, E)$ be a biconnected graph. For $S \subset V$ such that $G \setminus S$ is disconnected, $\sum_{i \in S} y_i \leq |S| - 1$ defines a facet of the maximum leaf spanning tree polytope of graph G if and only if*

1. *For any proper subset $S' \subsetneq S, G \setminus S'$ is connected; and*
2. *Let $G_1 = G[S_1], \ldots, G_\kappa = G[S_\kappa]$ be connected components of $G \setminus S$, where $S_1, \ldots, S_\kappa$ is a partition of $V \setminus S$. For $j \in V \setminus S$, $\sigma(j)$ is defined as the unique index k such that $j \in S_k$. Then, for any $j \in V \setminus S$, there exists $i \in S$ such that $G[\{i\} \cup S_{\sigma(j)} \setminus j]$ is connected.*

Proposition 6 below can be viewed as a translation of this result to the context of CDSs. Our characterization makes appropriate use of inclusion-wise minimality and r-robust vertex-cuts. Before stating the proposition, we require a lemma due to Kloks and Kratsch (1998). By *minimal vertex-cut*, we implicitly mean *inclusion-wise minimal vertex-cut*.

Lemma 4 (Kloks and Kratsch (1998))**.** *A vertex-cut $C \subset V$ of a graph $G = (V, E)$ is minimal if and only if every vertex of C has a neighbor in every connected component of $G[V \setminus C]$.*

Proposition 6. *Let $G = (V, E)$ be a biconnected graph and let C be a vertex-cut. The inequality $\sum_{i \in C} x_i \geq 1$ induces a facet of the 1-1-CDS polytope of G if and only if*

1. *C is a minimal (0-robust) vertex-cut, and*
2. *for every vertex $v \in V \setminus C$, the set $C \cup \{v\}$ is not a 1-robust vertex-cut.*

Proof. Proof. Validity is clear, so we provide n affinely independent points in the 1-1-CDS polytope where $\sum_{i \in C} x_i = 1$.

First we generate $|C|$ points. We claim that for every $v \in C$, the set $S_v = (V \setminus C) \cup \{v\}$ is a 1-1-CDS for G. First see that S_v is connected, since v has a neighbor in every connected component of $G[V \setminus C]$ by minimality of C and Lemma 4. Second, S_v is dominating, since every vertex not in S_v belongs to C, and every vertex in C has a neighbor in every connected component of $G[V \setminus C]$.

Now we give $n - |C|$ other points. Since for every vertex $v \in V \setminus C, C \cup \{v\}$ is not a 1-robust vertex-cut, this implies that for every $v \in V \setminus C$ there exists $w \in C$ such that $C \cup \{v\} \setminus \{w\}$ is not a vertex-cut. We claim that its complement $S_v = V \setminus ((C \cup \{v\}) \setminus \{w\})$ is a 1-1-CDS for G. We have established that $C \cup \{v\} \setminus \{w\}$ is not a vertex-cut, so its complement S_v induces a connected graph. Moreover, S_v is dominating. Every vertex $z \in C$ is dominated since it has a neighbor in every connected component of $G[V \setminus C]$, and there are at least two such components (so z remains dominated even if v is one of its neighbors). Every vertex in $V \setminus C$ (that is not v) belongs to S_v. Finally, v must be dominated, because: either (1) it is isolated in $G[V \setminus C]$ in which case by Lemma 4 it is adjacent to every vertex in C, implying that it is dominated; or (2) it is not isolated in $G[V \setminus C]$ in which case it has a neighbor in its connected component that belongs to S_v.

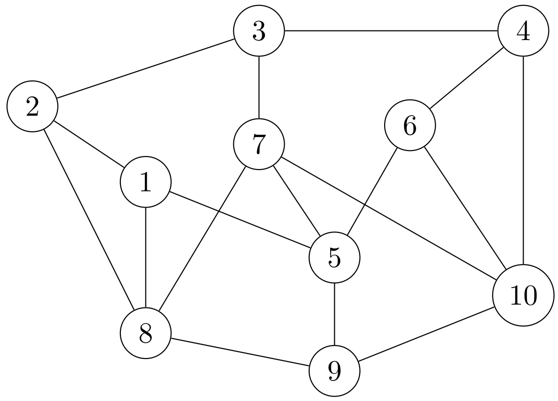
To prove 'only if', we will show that if condition 1 or 2 fails, then the vertex-cut inequality for C cannot induce a facet when the polytope is full-dimensional. Suppose that C is not minimal, implying that there exists $v \in C$ such that $C \setminus \{v\}$ is a vertex-cut. Then the inequality $\sum_{i \in C \setminus \{v\}} x_i \geq 1$ dominates the inequality $\sum_{i \in C} x_i \geq 1$. In the other case, C is minimal, yet there exists $v \in V \setminus C$ such that $C \cup \{v\}$ is a 1-robust vertex-cut. Then the 1-robust vertex-cut inequality $x_v + \sum_{i \in C} x_i \geq 2$ dominates the inequality $\sum_{i \in C} x_i \geq 1$. □

Unfortunately, given a vertex-cut, the problem of finding its 'robustness' – the largest value of r such that it is an r-robust vertex-cut – is coNP-hard. The associated decision problem is shown to be coNP-complete below. We note that this problem is polytime solvable when r is a constant; the naive algorithm runs in time $O(mn^r)$.

Proposition 7. *Given a graph $G = (V, E)$, a vertex-cut $C \subset V$, and a nonnegative integer r, the problem of determining if C is an r-robust vertex-cut is coNP-complete. This holds even when G is a split graph and C is a clique.*

Proof. Proof. The problem belongs to coNP, since a subset $C' \subseteq C$ of at most r vertices provides a short certificate that the answer is 'no' in the case that $C \setminus C'$ is not a vertex-cut for G. The reduction is from Set Cover defined by a ground set $\mathcal{U} = \{1, \dots, m\}$, a collection $\mathcal{S} = \{S_1, \dots, S_n\}$ of subsets of $\mathcal{U}$, and a nonnegative integer r. This problem asks: does there exist a subcollection $\mathcal{S}' \subset \mathcal{S}$ of r sets whose union is $\mathcal{U}$? We construct a graph $G = (V, E)$ and vertex-cut $C \subset V$ such that C is an r-robust vertex-cut if and only if the instance of Set Cover has **no** cover of size r.

Let $V = \mathcal{S} \cup \mathcal{U}$. Construct E by adding an edge between every pair of distinct vertices from $\mathcal{S}$, and for $i = 1, \dots, n$ add edges between the vertex S_i and




```
x1+ x2+ x3+ x4+x5+x6+ x7+x8+ x9+x10 >= 7
x1+ x2+2x3+ x4+x5+x6+ x7+x8    +x10 >= 7
x1+2x2    +2x4+x5+x6+2x7   +2x9     >= 7
x1+ x2    + x4   +x6+ x7   +2x9     >= 4
        x3    +x5        +x8    +x10 >= 3
x1+ x2    + x4      + x7   + x9     >= 3
x1    + x3       +x6+ x7   + x9     >= 3
    x2    + x4+x5   + x7   + x9     >= 3
    x2    + x4   +x6+ x7   + x9     >= 3
x1    + x3               +x8        >= 2
    x2    + x4+x5                   >= 2
    x2    + x4      + x7            >= 2
    x2        +x5        +x8        >= 2
        x3       +x6            +x10 >= 2
            x4+x5               +x10 >= 2
x1+ x2                   +x8+ x9     >= 2
        x3    +x5   + x7   + x9     >= 2
            x4   +x6       + x9+x10 >= 2
```

Figure 1: A graph and all facets of its 2-2-CDS polytope (excluding trivial 0-1 facets)

its elements from $\mathcal{U}$. Finally, let $C = \mathcal{S}$ be the vertex-cut. The set $\mathcal{S}$ is a clique and $\mathcal{U}$ is an independent set, so G is a split graph.

Suppose that C is not an r-robust vertex-cut. Then, there exists a subset $C' \subset C$ of $|C'| \leq r$ vertices such that $C \setminus C'$ is not a vertex-cut. In other words, $G[(V \setminus C) \cup C']$ is connected. This implies that C' is a cover of size at most r.

Now suppose that the instance of Set Cover has a cover $\mathcal{S}' \subset \mathcal{S} = C$ using only r sets from $\mathcal{S}$. Then $\mathcal{S}'$ is a 1-1-CDS for G of size r, implying that the inequality $\sum_{v \in C} x_v \geq 1 + r$ is not valid for the 1-1-CDS polytope of G. Hence, C cannot be an r-robust vertex-cut. □

We were unable to find similar good characterizations for when the r-robust vertex-cut inequalities induce facets of the k-d-CDS polytope of an arbitrary graph for values other than $d = k = 1$ and $r = 0$. However, through computational tests with PORTA (Christof et al. 1997), we were able to find examples showing that the r-robust vertex-cut inequalities, despite inducing exponentially many facets, do not fully describe the convex hull of integer feasible points. For example, Figure 1 provides the full description of the k-d-CDS polytope for a small graph. Another example showing that this holds even when $k = d = 1$ can be found in Appendix 5.

4 Solving the minimum k-d-CDS problem

In this section, we describe procedures for solving the minimum k-d-CDS problem and present computational results for $(k, d) = (1, 1), (2, 1), (2, 2), (3, 3), (4, 4)$.

The cases where $k = d$ are considered because they capture the notion of fault-tolerance—since a k-k-CDS remains a CDS if any fewer than k vertices fail. The reason for running tests with $(k, d) = (2, 1)$ is because this case has been considered in literature, e.g., by do Forte et al. (2013).

The formulations presented in Section 3 for the minimum k-d-CDS problem have exponentially many constraints, so it is not practical to include all such constraints a priori. Instead, we first define an initial set of constraints that enforces d-domination. Additional constraints that ensure that the selected vertices are k-connected are added as needed in a lazy fashion.

- **Initial constraints**: d-domination. For each vertex $v \in V$, we add the constraint $(d-k)x_v + \sum_{j \in N(v)} x_j \geq d$. When $d \neq k$, we have already described this inequality. When $d = k$, this inequality reduces to $\sum_{j \in N(v)} x_j \geq k$, which is valid since $N(v)$ is a vertex-cut.
- **Lazy constraints**: k-connectivity. Given an integer solution x' that is not k-connected, we want to find a vertex-cut constraint or an a-b-separator inequality that x' violates. Let $S \subset V$ denote the selected vertices in x'. First, find a minimum vertex-cut $C \subseteq S$ of $G[S]$. Then, the vertex subset $C' = C \cup (V \setminus S)$ is a vertex-cut for G. However, it may be rather large, so we use Algorithm 1 to find a minimal subet C'' of C' that remains a vertex-cut for G. Now, the constraint that we add will depend on the values of d and k.
  1. Case $d \geq k$. Add the inequality $\sum_{j \in C''} x_j \geq k$, which is valid since C'' is a vertex-cut.
  2. Case $d < k$. Here, C'' is an a-b-separator for some pair of vertices a and b, and the a-b-separator inequality $\sum_{j \in C''} x_j \geq k(x_a + x_b - 1)$ is valid. There can be many such inequalities depending on the choice of a and b, and it is an interesting question as to which and how many of them should be used. This is evaluated experimentally later in the paper.

Before adding a lazy constraint, one must first check if the subgraph induced by the current solution is k-connected. We rely upon existing agorithms. For general k, the fastest known algorithms to check if a graph is k-connected are as slow or slower than known max-flow algorithms. Specifically, the best known algorithm for computing unweighted vertex connectivity κ of an undirected graph runs in $O((n + \min\{\kappa^{5/2}, \kappa n^{3/4}\})\kappa n)$ time (Gabow 2006). However, for the particular cases $k = 1, 2, 3, 4$ that we consider, faster algorithms are known. Checking connectivity ($k = 1$) can be done in $O(m+n)$ time using BFS or DFS. All biconnected components ($k = 2$) and articulation vertices can be found in $O(m + n)$ as well (Tarjan 1972). Similarly triconnected components ($k = 3$) and 2-vertex-cuts can be found in $O(m + n)$ time (Hopcroft and Tarjan 1973). For the case $k = 4$, an $O(n^2)$ algorithm is known (Kanevsky and Ramachandran 1991). For $k = 1, 2, 3$ we use the $O(m + n)$ algorithms. For $k = 4$, we use an admittedly slower $O(mn)$ algorithm (instead of $O(n^2)$), which solves n

triconnectivity problems, for ease of implementation. We use the triconnectivity algorithm proposed by Hopcroft and Tarjan (1973), corrected by Gutwenger and Mutzel (2001), and implemented by Neumann (2011).

> **Data**: a vertex-cut $C \subset V$ and a graph $G = (V, E)$
> **Result**: an inclusion-wise minimal vertex-cut $C' \subseteq C$ for G
> $C' \leftarrow \{v \in C : \exists w \notin C \text{ with } \{v, w\} \in E\}$;
> $\mathbf{S} := \{S_1, \dots, S_p\} \leftarrow$ connected components of $G[V \setminus C']$, where $S_i \subset V \setminus C', i = 1, \dots, p$;
> **for** $v \in C'$ **do**
>   **if** *v has a neighbor in every connected component* **then**
>   `// Do nothing;` v `must remain in` C'`.`
>   **else**
>   Merge v and all components from $\mathbf{S}$ that v has a neighbor in;
> $C' \leftarrow C' \setminus \{v\}$;
>   **end**
> **end**
> **return** C'

Algorithm 1: Finding an (inclusion-wise) minimal vertex-cut in linear time

Proposition 8. *Algorithm 1 is correct and can be implemented to run in linear time.*

Proof. Proof. It is easy to see that C' and $\mathbf{S}$ can be constructed in time $O(m+n)$, e.g., by representing C' as a boolean n-vector and find $\mathbf{S}$ using BFS. There are $|C'| = O(n)$ iterations of the for-loop, and in each iteration we inspect the $|N(v)|$ neighbors of v, requiring at most $n + 2m$ operations. The merging operation for a vertex v can also be done in $|N(v)|$ time. Each vertex belonging to the same connected component will have a pointer to its component's identifier. Whenever merging v, loop across the vertices from $N(v)$, updating the identifier that its neighbors point to.

The algorithm certainly returns a vertex-cut; minimality is all that requires proof. See that if a vertex $v \in C'$ has a neighbor in every connected component in an iteration of the for-loop, it will continue to do so in subsequent iterations since the components from $\mathbf{S}$ are only merged—never split nor created. (Note that the first step of creating C' is done to ensure that 'merging' is well-defined.) Thus, every vertex $v \in C'$ has a neighbor in every connected component of $G[V \setminus C']$ and the returned vertex-cut is minimal by Lemma 4. $\square$

4.1 Computational setup and numerical experiments

All computational experiments were conducted on a *Dell Precision WorkStation T7500*® machine with two Intel Xeon® E5620 2.40 GHz quad-core processors and 12 GB RAM. The solver used was Gurobi Optimizer version 5.5 with its lazy-constraint callback (Gurobi Optimization, Inc 2013).

In Table 1, we compare the runtime of the lazy-constraint approach (referred to in the table as "Lazy") with other approaches for the MCDS problem. The first six approaches were proposed by Gendron et al. (2014), and include standalone (SA) and iterative-probing (IP) versions of Bender's decomposition (BE), branch-and-cut (BC), and a hybrid (HY) of BE and BC. The next three approaches are as follows: p-SABC is a branch-and-cut approach from Simonetti et al. (2011); DGR is a branch-and-cut approach from Lucena et al. (2010); and MTZ uses Miller-Tucker-Zemlin constraints to enforce connectivity (as proposed by Fan and Watson (2012)). We note that the experiments of Gendron et al. (2014) for the first six approaches and the MTZ approach were conducted on a 2.0 GHz Intel Xeon® E5405 machine with 8 GB RAM.

As can be seen from Table 1, the lazy-constraint approach seems to be more efficient than other approaches from literature. The comparisons are not entirely fair, since the computers and MIP solvers used are not the same. However, the drastic difference in runtimes suggests that the new approach is quicker, and not just because of better hardware. In fact, the lazy approach solved 42 of the 47 instances in under *10 seconds*, while no other approach was able to solve 42 instances in *one hour*. The instance v200_d10, which went unsolved by previous approaches, was solved in under 10 minutes by the lazy approach.

More details regarding the experiments for 1-1-CDS can be found in Tables 2 and 3. Table 2 includes the initial LP relaxation, the number of lazy cuts added, and the number of branch-and-bound nodes. Table 3 compares the computational effort expended when using arbitrary vertex-cuts versus minimal vertex-cuts. As can be seen from the tables, the most work is done whenever the graph is sparse, where the domination number and connected domination number tend to differ the most. Algorithm 1 becomes extremely important in these cases, solving several instances in a couple seconds that were otherwise unsolvable by the lazy approach.

In Table 4, we report experimental results for the minimum 2-1-CDS problem using the lazy approach. As mentioned in the previous section, a minimal vertex-cut C can be an a-b separator for many choices of a and b—leading to numerous possible lazy cuts. It was not clear to us which or how many of them should be used. This lead us to consider two possible approaches:

- (Single cut) add the a-b-separator inequality for which a and b are the lexicographically smallest, and
- (All cuts) add all possible a-b-separator inequalities.

As one might expect, using all cuts typically reduces the number of branch-and-bound nodes explored. However, this comes at the cost of solving a larger linear programming relaxation. In our tests, the single cut approach was generally faster, but not by much. In some cases, such as for the instance v200_d10, the single cut approach is significantly faster (3 minutes versus 50 minutes). In contrast, the all cuts approach was faster for the instance v200_d20 (1 minute versus 6 minutes). Still, the runtimes for the single cut approach are encouraging, as it solves each instance in less than 7 minutes. This is much quicker than other computational results for the minimum 2-1-CDS problem, such as by do Forte et al. (2013), in which most 150-vertex instances went unsolved in 2 hours. Unfortunately, we are unable to compare our runtimes on the same instances; do Forte et al. (2013) informed us that they could not locate the instances that they used in their paper.

In Table 5, we provide runtimes and solution sizes for the same instances for both the minimum k-total dominating set (kTDS) problem and the minimum k-k-CDS problem for $k = 1, 2, 3, 4$. Recall that the minimum k-total dominating set problem can be stated as

$$\text{(minimum } k\text{TDS problem)} \qquad \min_{x \in \{0,1\}^n} \left\{ \sum_{i \in V} x_i \;\middle|\; \sum_{j \in N(v)} x_j \geq k, \; \forall v \in V \right\}.$$

For most of the considered instances the times to solve minimum kTDS and minimum k-k-CDS are similar, meaning that the connectivity constraints need not add burden, in contrast with computational results from previous literature.

5 Conclusion

This paper studies a fault-tolerant connected dominating set (called a k-connected d-dominating set) and the associated minimization problem. We first identify what a k-d-CDS "looks like," allowing us to characterize precisely which graphs admit a k-d-CDS. Then we show the potential costliness of ensuring robustness; increasing the fault-tolerance parameters by a single unit can increase the size of an optimal solution from a small constant to nearly the entire vertex set. Three integer programming formulations for the minimum k-d-CDS problem are provided and their integer hulls are studied. The dimension of the k-d-CDS polytope can be easily identified whenever $d \geq k$, and we show precisely when the 0-1 bounds induce facets. We generalize previous inequalities from literature, resulting in what we call r-robust vertex-cut inequalities. These inequalities are shown to induce exponentially many facets. Then we consider solving the problem to optimality. A lazy-constraint integer programming approach is used to solve standard problem instances relatively quickly—in roughly the same time as the (not necessarily connected) dominating set problem. This is in contrast to previous approaches in literature where the connected versions of the problem took considerably longer to solve. In the particular case that $k = d = 1$ (i.e., the MCDS problem) a lazy-constraint approach is shown to be faster than previous approaches.

We finish by mentioning a few open problems.

- **Complexity in unit disk graphs**. It is straightforward to show that the minimum k-d-CDS problem remains hard for *any* fixed positive integers d and k in arbitrary graphs. However, in a significant number of applications, the problem instances are unit disk graphs. It is known that the MCDS problem remains hard in unit disk graphs (Lichtenstein 1982), but we are unaware of a similar result for k-d-CDS for $(k,d) \neq (1,1)$. Others, such as Thai et al. (2007), have stated that they expect it to remain hard for unit disk graphs.

- **Separating r-robust vertex-cut inequalities for $r > 0$.** The separation problem for the r-robust vertex-cut inequalities is polytime solvable when $r = 0$; however, we have not been able to establish the complexity for $r > 0$.
- **Conditions for r-robust vertex-cut inequalities to induce facets**. Proposition 6 states a good characterization for when r-robust vertex-cut inequalities induce facets of the k-d-CDS polytope when $k = d = 1$ and $r = 0$. We have not been able to find good characterizations for other values of k, d, and r.

Acknowledgements

This material is based upon work supported by the AFRL Mathematical Modeling and Optimization Institute. Partial support by AFOSR under grants FA9550-12-1-0103 and FA8651-12-2-0011 is also gratefully acknowledged.

References

Ahn, N., S. Park. 2014. An optimization algorithm for the minimum k-connected m-dominating set problem in wireless sensor networks. Tech. Rep. 2014-04, KAIST. URL `http://ie.kaist.ac.kr/isyse/professor/tech_file/techicalreports0605.pdf`.

Blum, J., M. Ding, A. Thaeler, X. Cheng. 2005. Connected dominating set in sensor networks and manets. *Handbook of Combinatorial Optimization* 329–369.

Buchanan, A., J.S. Sung, V. Boginski, S. Butenko. 2014. On connected dominating sets of restricted diameter. *European Journal of Operational Research* **236** 410–418.

Butenko, S., X. Cheng, C. Oliveira, P.M. Pardalos. 2004. A new heuristic for the minimum connected dominating set problem on ad hoc wireless networks. *Recent Developments in Cooperative Control and Optimization* 61–73.

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

Cheng, X., X. Huang, D. Li, W. Wu, D.Z. Du. 2003. A polynomial-time approximation scheme for the minimum-connected dominating set in ad hoc wireless networks. *Networks* **42** 202–208.

Christof, T., A. Löbel, M. Stoer. 1997. PORTA-polyhedron representation transformation algorithm. URL `http://www.iwr.uni-heidelberg.de/groups/comopt/software/PORTA/index.html`.

Dai, F., J. Wu. 2006. On constructing k-connected k-dominating set in wireless ad hoc and sensor networks. *Journal of Parallel and Distributed Computing* **66** 947–958.

do Forte, V.L., A. Lucena, N. Maculan. 2013. Formulations for the minimum 2-connected dominating set problem. *Electronic Notes in Discrete Mathematics* **41** 415–422.

Du, D.Z., P.J. Wan. 2013. *Connected Dominating Set: Theory and Applications*. Springer.

Fan, N., J.P. Watson. 2012. Solving the connected dominating set problem and power dominating set problem by integer programming. G. Lin, ed., *Combinatorial Optimization and Applications*, *Lecture Notes in Computer Science*, vol. 7402. Springer, 371–383.

Fomin, F.V., F. Grandoni, D. Kratsch. 2008. Solving connected dominating set faster than 2^n. *Algorithmica* **52** 153–166.

Fujie, T. 2003. An exact algorithm for the maximum leaf spanning tree problem. *Computers & Operations Research* **30** 1931–1944.

Fujie, T. 2004. The maximum-leaf spanning tree problem: Formulations and facets. *Networks* **43** 212–223.

Füredi, Z. 1987. The number of maximal independent sets in connected graphs. *Journal of Graph Theory* **11** 463–470.

Gabow, H.N. 2006. Using expander graphs to find vertex connectivity. *Journal of the ACM* **53** 800–844.

Gendron, B., A. Lucena, A.S. da Cunha, L. Simonetti. 2014. Benders decomposition, branch-and-cut, and hybrid algorithms for the minimum connected dominating set problem. *INFORMS Journal on Computing* To Appear.

Grötschel, M., L. Lovász, A. Schrijver. 1981. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica* **1** 169–197.

Guha, S., S. Khuller. 1998. Approximation algorithms for connected dominating sets. *Algorithmica* **20** 374–387.

Gurobi Optimization, Inc. 2013. Gurobi Optimizer Reference Manual. URL `http://www.gurobi.com`.

Gutwenger, C., P. Mutzel. 2001. A linear time implementation of SPQR-trees. J. Marks, ed., *Graph Drawing*, *Lecture Notes in Computer Science*, vol. 1984. Springer, 77–90.

Henzinger, M.R., S. Rao, H.N. Gabow. 2000. Computing vertex connectivity: New bounds from old techniques. *Journal of Algorithms* **34** 222–250.

Hopcroft, J.E., R.E. Tarjan. 1973. Dividing a graph into triconnected components. *SIAM Journal on Computing* **2** 135–158.

Hunt, H.B., M.V. Marathe, V. Radhakrishnan, S.S. Ravi, D.J. Rosenkrantz, R.E. Stearns. 1998. NC-approximation schemes for NP-and PSPACE-hard problems for geometric graphs. *Journal of Algorithms* **26** 238–274.

Kanevsky, A. 1990. On the number of minimum size separating vertex sets in a graph and how to find all of them. *Proceedings of the first annual ACM-SIAM Symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics, 411–421.

Kanevsky, A., V. Ramachandran. 1991. Improved algorithms for graph four-connectivity. *Journal of Computer and System Sciences* **42** 288–306.

Khachian, L.G. 1979. A polynomial algorithm in linear programming. *Soviet Mathematics Doklady* 191–194.

Kloks, T., D. Kratsch. 1998. Listing all minimal separators of a graph. *SIAM Journal on Computing* **27** 605–613.

Li, Y., Y. Wu, C. Ai, R. Beyah. 2012. On the construction of k-connected m-dominating sets in wireless networks. *Journal of Combinatorial Optimization* **23** 118–139.

Lichtenstein, D. 1982. Planar formulae and their uses. *SIAM Journal on Computing* **11** 329–343.

Lu, H.I., R. Ravi. 1998. Approximating maximum leaf spanning trees in almost linear time. *Journal of Algorithms* **29** 132–141.

Lucena, A., N. Maculan, L. Simonetti. 2010. Reformulations and solution algorithms for the maximum leaf spanning tree problem. *Computational Management Science* **7** 289–311.

Matula, D.W. 1978. k-blocks and ultrablocks in graphs. *Journal of Combinatorial Theory, Series B* **24** 1–13.

Morgan, M., V. Grout. 2008. Finding optimal solutions to backbone minimisation problems using mixed integer programming. *Proceedings of the 7th International Network Conference (INC 2008)*. 53–64.

Mycielski, J. 1955. Sur le coloriage des graphes. *Colloq. Math.* **3** 161–162.

Neumann, A. 2011. Implementation of Schmidts algorithm for certifying triconnectivity testing. Master's thesis, Saarland University.

Shang, W., P. Wan, F. Yao, X. Hu. 2007. Algorithms for minimum m-connected k-tuple dominating set problem. *Theoretical Computer Science* **381** 241–247.

Simonetti, L., A.S. da Cunha, A. Lucena. 2011. The minimum connected dominating set problem: formulation, valid inequalities and a branch-and-cut algorithm. *Network Optimization* 162–169.

Tarjan, R.E. 1972. Depth-first search and linear graph algorithms. *SIAM Journal on Computing* **1** 146–160.

Thai, M.T., N. Zhang, R. Tiwari, X. Xu. 2007. On approximation algorithms of k-connected m-dominating sets in disk graphs. *Theoretical Computer Science* **385** 49–59.

Trick, M. 2013. Graph coloring instances. URL `http://mat.gsia.cmu.edu/COLOR/instances.html`.

Weisstein, E. W. 2013. Perrin Sequence. *MathWorld*–A Wolfram Web Resource. URL `http://mathworld.wolfram.com/PerrinSequence.html`.

Wu, Y., Y. Li. 2008. Construction algorithms for k-connected m-dominating sets in wireless sensor networks. *Proceedings of the 9th ACM International Symposium on Mobile Ad Hoc Networking and Computing*. ACM, 83–90.

Yuan, D. 2005. Energy-efficient broadcasting in wireless ad hoc networks: performance benchmarking and distributed algorithms based on network connectivity characterization. *Proceedings of the 8th ACM International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems*. ACM, 28–35.

Zhang, Z., X. Gao, W. Wu, D.Z. Du. 2008. PTAS for minimum connected dominating set in unit ball graph. *Wireless Algorithms, Systems, and Applications* 154–161.

Zhou, J., Z. Zhang, W. Wu, K. Xing. 2014. A greedy algorithm for the fault-tolerant connected dominating set in a general graph. *Journal of Combinatorial Optimization* **28** 310–319.

Appendices

CDS facets for the Grötzsch graph

The convex hull of characteristic vectors of 1-1-CDSs of the Mycielski graph (Mycielski 1955) on eleven vertices (also known as the Grötzsch graph) is not fully described by the r-robust vertex-cut inequalities. Below are all facet-defining inequalities – except for the trivial 0-1 facets – as given by PORTA (Christof et al. 1997). The vertex numbering is the same as in the graph file provided by Michael Trick for the DIMACS coloring challenge (Trick 2013).

```
 x1+ x2+ x3+ x4+ x5+ x6+ x7+ x8+ x9+ x10+ x11 >= 4
 x1+ x2+ x3+ x4+ x5+ x6    + x8         +2x11 >= 3
 x1+ x2+ x3+ x4+ x5+ x6            + x10+2x11 >= 3
 x1+ x2+ x3+ x4+ x5    + x7    + x9     +2x11 >= 3
 x1+ x2+ x3+ x4+ x5    + x7        + x10+2x11 >= 3
 x1+ x2+ x3+ x4+ x5        + x8+ x9     +2x11 >= 3
2x1    + x3    + x5+2x6+ x7+ x8+ x9+ x10+ x11 >= 3
 x1+ x2        +2x5+ x6+ x7+ x8+ x9+2x10+ x11 >= 3
 x1    +2x3+ x4    + x6+ x7+2x8+ x9+ x10+ x11 >= 3
    2x2    + x4+ x5+ x6+2x7+ x8+ x9+ x10+ x11 >= 3
     x2+ x3+2x4    + x6+ x7+ x8+2x9+ x10+ x11 >= 3
 x1+ x2+ x3+ x4+ x5                     +3x11 >= 3
 x1+ x2+ x3+ x4            + x8+ x9     + x11 >= 2
 x1+ x2+ x3    + x5+ x6            + x10+ x11 >= 2
 x1+ x2    + x4+ x5    + x7        + x10+ x11 >= 2
 x1    + x3+ x4+ x5+ x6    + x8         + x11 >= 2
     x2+ x3+ x4+ x5    + x7    + x9     + x11 >= 2
 x1+ x2        + x5+ x6+ x7+ x8+ x9+2x10      >= 2
 x1    + x3+ x4    + x6+ x7+2x8+ x9+ x10      >= 2
 x1    + x3    + x5+2x6+ x7+ x8+ x9+ x10      >= 2
     x2+ x3+ x4    + x6+ x7+ x8+2x9+ x10      >= 2
     x2    + x4+ x5+ x6+2x7+ x8+ x9+ x10      >= 2
 x1    + x3                             + x11 >= 1
 x1            + x5                     + x11 >= 1
     x2    + x4                         + x11 >= 1
     x2        + x5                     + x11 >= 1
         x3+ x4                         + x11 >= 1
 x1    + x3        + x6    + x8               >= 1
 x1            + x5+ x6            + x10      >= 1
     x2    + x4        + x7    + x9           >= 1
     x2        + x5    + x7        + x10      >= 1
         x3+ x4            + x8+ x9           >= 1
                     x6+ x7+ x8+ x9+ x10      >= 1
```

Table 1: A comparison of runtimes, in seconds, to solve the MCDS problem (i.e., minimum 1-1-CDS). Aside from the “Lazy” approach, all times are from Gendron et al. (2014). A dash indicates unsolved in time limit.

Instance	SABE	IPBE	SABC	IPBC	SAHY	IPHY	p-SABC	DGR	MTZ	Lazy
v30_d10	1222.10	756.87	0.03	0.02	6.11	2.88	0.01	0.01	36.20	0.24
v30_d20	0.01	0.00	0.02	0.02	0.01	0.02	0.02	0.10	0.92	0.01
v30_d30	0.02	0.02	0.05	0.06	0.03	0.02	0.05	0.03	1.23	0.01
v30_d50	0.00	0.00	0.01	0.03	0.01	0.01	0.04	0.08	1.07	0.01
v30_d70	0.00	0.00	0.02	0.00	0.01	0.00	0.02	0.01	0.07	0.01
v50_d5	-	-	0.01	0.02	88.63	9.46	0.02	0.01	182.06	0.59
v50_d10	34.30	3.09	0.82	0.20	2.69	3.89	0.42	0.36	5.18	0.12
v50_d20	0.21	0.09	0.77	0.97	0.40	0.16	0.66	1.32	4.23	0.08
v50_d30	0.18	0.11	0.32	0.25	0.45	0.31	0.25	1.21	14.90	0.07
v50_d50	0.00	0.01	0.23	0.06	0.01	0.01	0.25	0.51	4.80	0.01
v50_d70	0.00	0.00	0.24	0.01	0.01	0.01	0.29	0.04	0.74	0.02
v70_d5	-	-	2.06	0.39	188.65	674.75	1.42	0.26	2098.18	1.41
v70_d10	1.06	2.17	18.68	5.25	25.16	1.26	34.29	4.73	18.03	0.09
v70_d20	0.38	0.17	2.68	1.88	1.15	0.58	2.16	16.30	72.49	0.15
v70_d30	0.54	0.21	1.20	0.99	0.82	0.37	1.00	2.90	54.98	0.17
v70_d50	0.01	0.02	0.64	0.40	0.02	0.02	0.70	1.33	5.64	0.01
v70_d70	0.00	0.01	0.99	0.04	0.02	0.02	0.79	1.92	15.27	0.07
v100_d5	-	-	58.77	64.13	-	142.49	342.25	12.50	872.93	0.36
v100_d10	0.49	0.33	28.25	39.71	2.68	1.70	32.11	9.36	176.67	0.34
v100_d20	1.88	1.26	283.23	414.49	6.48	2.70	174.93	86.16	460.49	0.40
v100_d30	3.83	2.46	329.05	638.89	11.17	4.42	193.65	258.15	1462.83	0.94
v100_d50	1.55	0.76	48.00	41.51	3.23	1.56	35.41	132.55	101.29	0.70
v100_d70	1.55	0.03	13.20	12.02	1.57	0.91	12.03	154.10	50.68	1.27
v120_d5	3.36	18.16	1465.05	199.01	102.61	35.10	-	2.65	258.56	0.31
v120_d10	23.97	3.86	-	-	56.31	18.68	-	65.49	178.19	0.34
v120_d20	5.02	3.79	1316.70	-	16.47	8.31	610.89	393.47	1967.54	1.86
v120_d30	5.25	4.44	790.91	1913.36	14.21	7.56	475.54	653.70	2241.50	2.32
v120_d50	4.21	2.52	246.93	202.30	8.73	4.57	168.55	815.64	145.95	1.64
v120_d70	2.25	0.04	36.84	28.90	2.82	2.22	31.67	356.31	80.97	2.44
v150_d5	-	771.07	-	-	-	-	-	2954.00	-	3.46
v150_d10	51.09	28.28	-	-	652.35	195.76	-	3247.89	-	4.72
v150_d20	367.30	271.65	-	-	2116.24	903.76	-	-	-	9.34
v150_d30	21.12	11.25	2972.83	-	34.61	24.77	1954.00	2317.35	-	6.54
v150_d50	7.81	5.78	724.92	477.10	17.57	10.79	481.61	2756.36	257.47	2.41
v150_d70	4.30	0.06	62.56	49.69	5.01	2.95	43.75	1828.86	133.09	4.77
v200_d5	-	1658.85	-	-	-	-	-	-	-	32.92
v200_d10	-	-	-	-	-	-	-	-	-	496.43
v200_d20	1686.30	1945.80	-	-	-	-	-	-	-	243.25
v200_d30	3210.83	1847.88	-	-	-	-	-	-	-	172.55
v200_d50	24.79	19.33	3363.33	1887.43	44.42	28.54	2249.43	20155.00	3509.21	8.16
v200_d70	10.53	0.13	340.20	275.84	9.17	5.63	271.91	8154.13	507.44	9.45
IEEE-14-Bus	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.01	0.20	0.01
IEEE-30-Bus	0.00	0.00	0.01	0.01	0.00	0.01	0.01	0.01	0.52	0.01
IEEE-57-Bus	-	-	0.02	0.03	24.40	192.43	0.00	0.16	-	1.07
RTS96	-	-	2.56	4.22	118.53	26.56	0.35	1.66	-	0.69
IEEE-118-Bus	-	-	0.22	0.51	65.65	2.88	0.07	0.36	-	0.08
IEEE-300-Bus	-	-	-	-	-	-	-	1076.89	-	52.88
# in 1 sec	15	20	17	19	13	15	19	14	5	27
# in 10 sec	27	29	21	22	24	30	21	23	11	42
# in 100 sec	33	33	28	28	35	36	27	27	19	44
# in 1000 sec	34	36	34	34	39	41	35	34	30	47
# solved	37	39	38	36	40	41	37	40	35	47
# fastest	8	11	1	2	2	2	5	3	0	28

Table 2: Extended computational results for solving the minimum 1-1-CDS problem using the lazy approach.

Instance	IP Opt	Root LP	# Lazy cuts	# B&B nodes	IP Time
v30_d10	15	12.00	32	82	0.24
v30_d20	7	5.87	1	0	0.01
v30_d30	4	3.31	3	0	0.01
v30_d50	3	2.01	0	0	0.01
v30_d70	2	1.43	0	0	0.01
v50_d5	31	21.00	348	4,356	0.59
v50_d10	12	10.23	14	96	0.12
v50_d20	7	5.21	2	41	0.08
v50_d30	5	3.46	0	13	0.07
v50_d50	3	2.05	0	0	0.01
v50_d70	2	1.46	0	0	0.02
v70_d5	27	21.75	194	3,398	1.41
v70_d10	13	11.05	11	51	0.09
v70_d20	7	5.29	4	119	0.15
v70_d30	5	3.37	0	26	0.17
v70_d50	3	2.08	0	0	0.01
v70_d70	2	1.45	0	0	0.07
v100_d5	24	20.43	51	1,273	0.36
v100_d10	13	10.86	10	680	0.34
v100_d20	8	5.35	0	546	0.40
v100_d30	6	3.43	0	1,351	0.94
v100_d50	4	2.12	0	237	0.70
v100_d70	3	1.46	0	118	1.27
v120_d5	25	22.35	13	643	0.31
v120_d10	13	10.51	10	244	0.34
v120_d20	8	5.29	0	2,897	1.86
v120_d30	6	3.43	0	2,642	2.32
v120_d50	4	2.02	1	442	1.64
v120_d70	3	1.44	0	151	2.44
v150_d5	26	21.48	39	12,371	3.46
v150_d10	14	10.77	5	8,355	4.72
v150_d20	9	5.20	4	37,253	9.34
v150_d30	6	3.48	1	3,840	6.54
v150_d50	4	2.00	0	670	2.41
v150_d70	3	1.45	0	183	4.77
v200_d5	27	22.39	47	146,750	32.92
v200_d10	16	10.56	8	2,182,664	496.43
v200_d20	9	5.02	1	190,963	243.25
v200_d30	7	3.37	1	287,638	172.55
v200_d50	4	2.02	0	1,234	8.16
v200_d70	3	1.45	0	271	9.45
IEEE-14-Bus	5	4.75	1	0	0.01
IEEE-30-Bus	11	10.00	5	0	0.01
IEEE-57-Bus	31	21.67	844	8,457	1.07
RTS96	32	24.50	452	7,825	0.69
IEEE-118-Bus	43	39.00	30	148	0.08
IEEE-300-Bus	129	98.50	10,628	103,314	52.88

Table 3: The effect of Algorithm 1 on reducing the computational effort for the minimum 1-1-CDS problem.

	Without Algorithm 1			With Algorithm 1		
Instance	# Lazy cuts	# B&B nodes	Time	# Lazy cuts	# B&B nodes	Time
v30_d10	54,750	81,015	770.39	32	82	0.24
v30_d20	3	0	0.02	1	0	0.01
v30_d30	3	0	0.02	3	0	0.01
v30_d50	0	0	0.00	0	0	0.01
v30_d70	0	0	0.00	0	0	0.01
v50_d5	>110,425	>125,657	>3600.00	348	4,356	0.59
v50_d10	91	297	0.55	14	96	0.12
v50_d20	4	29	0.08	2	41	0.08
v50_d30	0	13	0.08	0	13	0.07
v50_d50	0	0	0.02	0	0	0.01
v50_d70	0	0	0.02	0	0	0.02
v70_d5	>88,233	>97,749	>3600.00	194	3,398	1.41
v70_d10	32	179	0.11	11	51	0.09
v70_d20	3	177	0.13	4	119	0.15
v70_d30	0	26	0.16	0	26	0.17
v70_d50	0	0	0.02	0	0	0.01
v70_d70	0	0	0.08	0	0	0.07
v100_d5	10,266	14,891	104.23	51	1,273	0.36
v100_d10	10	211	0.25	10	680	0.34
v100_d20	0	546	0.41	0	546	0.40
v100_d30	0	1,351	0.92	0	1,351	0.94
v100_d50	0	237	0.70	0	237	0.70
v100_d70	0	118	1.23	0	118	1.27
v120_d5	173	1,071	0.67	13	643	0.31
v120_d10	14	220	0.31	10	244	0.34
v120_d20	0	2,897	1.86	0	2,897	1.86
v120_d30	0	2,642	2.25	0	2,642	2.32
v120_d50	1	465	1.20	1	442	1.64
v120_d70	0	151	2.48	0	151	2.44
v150_d5	1,125	15,506	32.03	39	12,371	3.46
v150_d10	7	13,094	4.70	5	8,355	4.72
v150_d20	4	38,251	10.00	4	37,253	9.34
v150_d30	1	5,120	9.35	1	3,840	6.54
v150_d50	0	670	2.50	0	670	2.41
v150_d70	0	183	4.65	0	183	4.77
v200_d5	138	199,543	84.20	47	146,750	32.92
v200_d10	6	2,220,011	520.21	8	2,182,664	496.43
v200_d20	1	192,712	240.82	1	190,963	243.25
v200_d30	1	287,645	172.99	1	287,638	172.55
v200_d50	0	1,234	8.10	0	1,234	8.16
v200_d70	0	271	9.45	0	271	9.45
IEEE-14-Bus	1	0	0.03	1	0	0.01
IEEE-30-Bus	47	54	0.05	5	0	0.01
IEEE-57-Bus	>102,288	>116,896	>3600.00	844	8,457	1.07
RTS96	>78,608	>77,532	>3600.00	452	7,825	0.69
IEEE-118-Bus	>67,771	>43,083	>3600.00	30	148	0.08
IEEE-300-Bus	>39,576	>86,494	>3600.00	10,628	103,314	52.88

Table 4: Comparing two approaches for solving minimum 2-1-CDS: adding a single lazy cut versus adding all lazy cuts.

	IP Opt	Single cut			All cuts		
		# Lazy Cuts	# B&B nodes	Time	# Lazy Cuts	# B&B nodes	Time
v30_d10	18	1	0	0.02	60	0	0.03
v30_d20	8	10	18	0.04	105	10	0.04
v30_d30	5	0	0	0.02	0	0	0.02
v30_d50	3	0	0	0.01	0	0	0.01
v30_d70	3	0	21	0.05	0	21	0.05
v50_d5	∞	0	0	0.00	0	0	0.00
v50_d10	14	4	3	0.05	122	5	0.11
v50_d20	7	0	0	0.04	0	0	0.04
v50_d30	5	0	0	0.09	0	0	0.09
v50_d50	3	0	0	0.08	0	0	0.09
v50_d70	3	0	32	0.08	0	32	0.08
v70_d5	34	0	0	0.01	0	0	0.01
v70_d10	14	9	197	0.10	433	145	0.35
v70_d20	8	0	119	0.13	0	119	0.14
v70_d30	5	2	284	0.21	28	177	0.17
v70_d50	3	0	0	0.08	0	0	0.09
v70_d70	3	0	69	0.16	0	69	0.17
v100_d5	28	12	540	0.20	798	504	0.52
v100_d10	14	6	562	0.29	314	532	0.70
v100_d20	8	0	1,101	0.73	0	1,101	0.76
v100_d30	6	0	1,505	1.00	0	1,505	1.01
v100_d50	4	0	98	0.53	0	98	0.56
v100_d70	3	0	110	1.34	0	110	1.37
v120_d5	27	1	216	0.17	68	216	0.16
v120_d10	14	1	643	0.60	40	256	0.47
v120_d20	8	2	9,250	3.83	24	7,479	5.39
v120_d30	6	1	1,106	1.22	10	1,295	2.40
v120_d50	4	0	168	3.29	0	168	3.26
v120_d70	3	0	127	0.52	0	127	0.52
v150_d5	28	6	2,707	0.93	975	2,675	8.43
v150_d10	15	1	8,833	3.94	30	8,281	4.68
v150_d20	9	1	15,744	6.56	24	15,744	6.44
v150_d30	6	0	2,665	2.64	0	2,665	2.65
v150_d50	4	0	233	3.10	0	233	2.88
v150_d70	3	0	189	4.81	0	189	4.83
v200_d5	29	9	128,408	27.47	584	122,234	197.13
v200_d10	16	4	308,635	172.73	303	3,328,419	3061.52
v200_d20	9	1	1,296,791	386.55	15	120,163	64.35
v200_d30	7	0	163,791	157.67	0	163,791	158.39
v200_d50	4	0	298	8.42	0	298	8.42
v200_d70	3	0	269	9.63	0	269	9.48
IEEE-14	7	4	0	0.02	34	0	0.01
IEEE-30	15	0	0	0.01	0	0	0.00
IEEE-57	37	1	0	0.01	99	0	0.01
RTS-96	38	200	957	0.16	36,555	366	5.91
IEEE-118	∞	0	0	0.01	0	0	0.01
IEEE-300	∞	0	0	0.01	0	0	0.01

Table 5: A comparison of running times for minimum k-total dominating set and minimum k-k-CDS. Blank entries denote that the instance is infeasible.

	1TDS		1-1-CDS		2TDS		2-2-CDS		3TDS		3-3-CDS		4TDS		4-4-CDS	
Graph	Opt	Time	Opt	Time	Opt	Time	Opt	Time	Opt	Time	Opt	Time	Opt	Time	Opt	Time
v30_d10	12	0.01	15	0.24												
v30_d20	6	0.01	7	0.01												
v30_d30	4	0.01	4	0.01	8	0.05	8	0.06	11	0.00	11	0.05	15	0.02	15	0.02
v30_d50	3	0.01	3	0.01	5	0.02	5	0.00	7	0.02	7	0.00	9	0.02	9	0.02
v30_d70	2	0.01	2	0.01	4	0.03	4	0.03	5	0.02	5	0.02	6	0.06	6	0.05
v50_d5	21	0.01	31	0.59												
v50_d10	11	0.01	12	0.12	22	0.00	22	0.00								
v50_d20	7	0.10	7	0.08	12	0.08	12	0.08	17	0.02	17	0.02				
v50_d30	5	0.18	5	0.07	8	0.20	8	0.06	12	0.08	12	0.28	15	0.03	15	0.09
v50_d50	3	0.01	3	0.01	5	0.06	5	0.05	7	0.02	7	0.05	9	0.02	9	0.02
v50_d70	2	0.01	2	0.02	4	0.05	4	0.09	5	0.02	5	0.05	7	0.14	7	0.11
v70_d5	23	0.01	27	1.41	47	0.00	47	0.00								
v70_d10	13	0.18	13	0.09	23	0.02	24	0.09								
v70_d20	7	0.13	7	0.15	12	0.13	12	0.14	17	0.23	17	0.20	23	0.13	23	0.13
v70_d30	5	0.15	5	0.17	8	0.19	8	0.20	12	0.19	12	0.19	15	0.14	15	0.14
v70_d50	3	0.02	3	0.01	5	0.23	5	0.20	7	0.19	7	0.13	9	0.23	9	0.20
v70_d70	2	0.07	2	0.07	4	0.36	4	0.36	5	0.17	5	0.50	7	0.27	7	0.39
v100_d5	23	0.11	24	0.36	44	0.08	44	0.08								
v100_d10	13	0.26	13	0.34	24	0.27	24	0.48	35	0.25	35	0.27				
v100_d20	8	0.47	8	0.40	13	0.45	13	0.45	18	0.27	18	0.34	23	0.48	23	0.50
v100_d30	6	0.72	6	0.94	9	0.78	9	1.89	13	2.34	13	2.48	16	1.12	16	1.22
v100_d50	4	0.67	4	0.70	6	0.50	6	0.62	8	0.69	8	0.81	10	0.58	10	0.64
v100_d70	3	1.07	3	1.27	4	0.55	4	0.58	6	0.58	6	0.67	7	0.97	7	0.69
v120_d5	25	0.21	25	0.31	46	0.16	46	0.16								
v120_d10	13	0.41	13	0.34	24	1.11	24	0.98	35	1.84	35	1.97	46	0.69	46	0.94
v120_d20	8	2.63	8	1.86	13	2.04	13	2.48	18	1.09	18	6.72	23	1.40	23	1.19
v120_d30	6	2.43	6	2.32	9	3.01	9	3.03	12	5.16	12	2.84	16	5.20	16	6.38
v120_d50	4	1.30	4	1.64	6	1.76	6	1.81	8	2.17	8	2.31	10	1.89	10	1.95
v120_d70	3	2.12	3	2.44	4	1.62	4	1.94	6	1.34	6	2.76	7	0.97	7	0.92
v150_d5	24	0.49	26	3.46	45	0.31	45	0.34	71	0.28	71	0.28				
v150_d10	14	2.91	14	4.72	24	2.92	24	6.19	35	5.40	35	6.71	46	31.01	46	31.12
v150_d20	8	2.03	9	9.34	14	27.55	14	28.11	19	79.65	19	80.03	24	123.77	24	121.24
v150_d30	6	5.31	6	6.54	10	17.29	10	14.10	13	136.02	13	56.01	17	320.29	17	329.66
v150_d50	4	2.46	4	2.41	6	3.11	6	3.23	8	5.04	8	2.40	10	3.35	10	3.32
v150_d70	3	3.74	3	4.77	4	4.65	4	4.93	6	2.54	6	3.20	7	2.29	7	2.31
v200_d5	27	11.49	27	32.92	48	10.41	48	9.36	71	4.35	71	5.24				
v200_d10	16	775.14	16	496.43	26	2,018.38	26	2,111.52	36	3,026.18	36	3,103.52	46	2,846.70	46	1,802.14
v200_d20	9	230.20	9	243.25	14	739.68	14	832.97	19	1,383.64	19	21,766.20	24	4,074.83	24	4,171.82
v200_d30	6	162.12	7	172.55	10	227.16	10	315.61	14	5,280.93	14	5,623.75	17	3,936.37	17	4,040.89
v200_d50	4	8.41	4	8.16	6	40.14	6	33.34	8	418.34	8	246.05	11	1,008.54	11	1,063.84
v200_d70	3	9.31	3	9.45	5	44.41	5	35.63	6	8.28	6	9.08	8	136.03	8	132.24