



# Enumerating Minimal Connected Dominating Sets

Faisal N. Abu-Khzam  

Department of Computer Science and Mathematics, Lebanese American University, Beirut, Lebanon

Henning Fernau   

Fachbereich IV, Informatikwissenschaften, Universität Trier, Germany

Benjamin Gras  

Université d'Orléans, INSA Centre Val de Loire, LIFO EA 4022, France

Mathieu Liedloff  

Université d'Orléans, INSA Centre Val de Loire, LIFO EA 4022, France

Kevin Mann  

Fachbereich IV, Informatikwissenschaften, Universität Trier, Germany

---

## Abstract

---

The question to enumerate all (inclusion-wise) minimal connected dominating sets in a graph of order  $n$  in time significantly less than  $2^n$  is an open question that was asked in many places. We answer this question affirmatively, by providing an enumeration algorithm that runs in time  $\mathcal{O}(1.9896^n)$ , using polynomial space only. The key to this result is the consideration of this enumeration problem on 2-degenerate graphs, which is proven to be possible in time  $\mathcal{O}(1.9767^n)$ . Apart from solving this old open question, we also show new lower bound results. More precisely, we construct a family of graphs of order  $n$  with  $\Omega(1.4890^n)$  many minimal connected dominating sets, while previous examples achieved  $\Omega(1.4422^n)$ . Our example happens to yield 4-degenerate graphs. Additionally, we give lower bounds for the previously not considered classes of 2-degenerate and of 3-degenerate graphs, which are  $\Omega(1.3195^n)$  and  $\Omega(1.4723^n)$ , respectively.

We also address essential questions concerning output-sensitive enumeration. Namely, we give reasons why our algorithm cannot be turned into an enumeration algorithm that guarantees polynomial delay without much efforts. More precisely, we prove that it is NP-complete to decide, given a graph  $G$  and a vertex set  $U$ , if there exists a minimal connected dominating set  $D$  with  $U \subseteq D$ , even if  $G$  is known to be 2-degenerate. Our reduction also shows that even any subexponential delay is not easy to achieve for enumerating minimal connected dominating sets. Another reduction shows that no FPT-algorithms can be expected for this extension problem concerning minimal connected dominating sets, parameterized by  $|U|$ . This also adds one more problem to the still rather few natural parameterized problems that are complete for the class W[3]. We also relate our enumeration problem to the famous open HITTING SET TRANSVERSAL problem, which can be phrased in our context as the question to enumerate all minimal dominating sets of a graph with polynomial delay by showing that a polynomial-delay enumeration algorithm for minimal connected dominating sets implies an affirmative algorithmic solution to the HITTING SET TRANSVERSAL problem.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Parameterized complexity and exact algorithms

**Keywords and phrases** enumeration problems, connected domination, degenerate graphs

**Digital Object Identifier** 10.4230/LIPIcs.ESA.2022.1

**Related Version** *Full Version:* <https://doi.org/10.48550/arXiv.2205.00086> [1]

**Acknowledgements** Henning Fernau likes to thank the Université d'Orléans for inviting him as a *professeur invité* in 2021; on these visits, much of the research was started.



© Faisal N. Abu-Khzam, Henning Fernau, Benjamin Gras, Mathieu Liedloff, and Kevin Mann; licensed under Creative Commons License CC-BY 4.0

30th Annual European Symposium on Algorithms (ESA 2022).

Editors: Shiri Chechik, Gonzalo Navarro, Eva Rotenberg, and Grzegorz Herman; Article No. 1; pp. 1:1–1:15

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

## 1 Introduction

The enumeration of objects that satisfy a given property has applications in many scientific domains including biology and artificial intelligence. Enumeration can also be used as part of an exact algorithm, e.g., confer the algorithm by Lawler [27] to compute a coloring of an input graph using a minimum number of colors. The dynamic programming scheme used by this algorithm needs all the maximal independent sets of the input graph. It is worth noting that the running time depends mainly on a bound on the number of maximal independent sets as well as on the running time of an algorithm that would produce all these sets.

Clearly, the number of outputs of an enumeration algorithm can be exponential in the size of the given input. It is the case for the number of maximal independent sets: there are graphs with  $3^{n/3}$  such sets [30], where  $n$  is the number of vertices in the graph. The running time of enumeration algorithms can either be measured with respect to the size of the input plus the size of the outputted set of objects, which is called *output-sensitive* analysis, or it can be measured according to the size of the input only, being called *input-sensitive* analysis. In the latter, the running time upper bound often implies an upper bound on the number of enumerated objects, *i.e.*, the maximum number of objects that can fulfill the given property.

Given a graph  $G = (V, E)$ , the problem of computing a minimum dominating set asks for a smallest-cardinality subset  $S \subseteq V$  such that each vertex not in  $S$  has at least one neighbor in  $S$ . This well studied NP-hard problem attracted considerable attention for decades. Several exponential-time algorithms have been designed to solve the problem exactly, and the most recent are based on *Measure-and-Conquer* techniques to analyze their running times [15, 24, 31]. The problem of enumerating all inclusion-minimal dominating sets has also caught attention for general graphs as well as for special graph classes [10, 16, 19].

Many variants of the dominating set problem have also gained attention [22]. In particular, the minimum *connected* dominating set problem requires that the graph induced by  $S$  is connected. The problem has attracted great attention and various methods have been devised to solve it exactly [3, 14]. A more challenging question has been posed about the enumeration of inclusion-minimal connected dominating sets. Already designing an algorithm faster than  $\text{poly}(n)2^n$  is known to be challenging, and this specific question has been asked several times as an open problem [6, 13, 20]. A recent result by Lokshantov et al. [29] shows that minimal connected dominating sets can be enumerated in time  $2^{(1-\epsilon)n} \cdot n^{\mathcal{O}(1)}$ , which broke the  $2^n$ -barrier for the first time. It is worth noting that  $\epsilon$  is a tiny constant, around  $10^{-50}$ , and it has remained open whether an algorithm exists that can substantially break the  $2^n$ -barrier. The enumeration of minimal connected dominating sets also received notable interest when the input is restricted to special graph classes [20, 21, 32, 33, 34].

On the other hand, the maximum number of minimal CDS in a graph was shown to be in  $\Omega(3^{\frac{n}{3}})$  [20], which is obviously very low compared to the currently best upper bound. This gap between upper and lower bounds is narrower when it comes to special graph classes. On chordal graphs, for example, the upper bound has been recently improved to  $\mathcal{O}(1.4736^n)$  [21]. Other improved lower/upper bounds have been obtained for AT-free, strongly chordal, distance-hereditary graphs, and cographs in [20]. Further improved bounds for split graphs, cobipartite and convex bipartite graphs have been obtained in [34] and [33]. Moreover, although the optimization problem seems simpler, the best-known exact algorithm solves the problem in time  $\mathcal{O}(1.8619^n)$  [3]. This is already much larger than the best-known lower bounds of  $3^{(n-2)/3}$  [20] to enumerate all minimal connected dominated sets.

In this paper, we show that the enumeration of all inclusion-wise minimal connected dominating sets can be achieved in time  $\mathcal{O}(1.9896^n)$ . Surprisingly, achieving this improvement was simply based on first considering the same enumeration on 2-degenerate graphs and

proving it to be possible in time  $\mathcal{O}(1.9767^n)$ . Achieving enumeration with polynomial delay is believed to be hard, since it would also lead to the same for the enumeration of minimal dominating sets, which has been open for several decades. We give further evidence of this (possible) hardness by showing that extending a subset of vertices into a minimal connected dominating set is NP-complete and also hard in a natural parameterized setting. Furthermore, we narrow the gap between upper and lower bounds by showing that the maximum number of minimal connected dominating sets in a graph is in  $\Omega(1.4890^n)$ , thus improving the previous lower bound of  $\Omega(1.4422^n)$ . Our construction yields new lower bounds on several special graph classes such as 3-degenerate planar bipartite graphs. For space restrictions, most proofs are either omitted or reduced to proof sketches; full proofs can be found in the long version of this paper [1].

## 2 Definitions, Preliminaries and Summary of Main Results

In this paper, we deal with undirected simple finite graphs that can be specified as  $G = (V, E)$ , where  $V$  is the finite vertex set and  $E \subseteq \binom{V}{2}$  is the set of edges. The number of vertices  $|V|$  is also called the *order* of graph  $G$  and is denoted by  $n$ . An edge  $\{u, v\}$  is usually written as  $uv$ . Alternatively,  $E$  can be viewed as a symmetric binary relation, so that  $E^*$  is then the transitive closure of  $E$ , which is an equivalence relation whose equivalence classes are also known as *connected components*. A graph is called *connected* if it has only one connected component. A graph  $G' = (V', E')$  is a *subgraph* of  $G = (V, E)$  if  $V' \subseteq V$  and  $E' \subseteq E$ ;  $G'$  is a *partial graph* of  $G$  if  $V = V'$ . A set of vertices  $S$  *induces* the subgraph  $G[S] = (S, E_S)$ , where  $E_S = \{uv \in E \mid u, v \in S\}$ ;  $S$  is called *connected* if  $G[S]$  is connected. For a vertex  $v \in V$ ,  $N_G(v) = \{u \in V \mid uv \in E\}$  is the *open neighborhood* of  $v$ , collecting the vertices *adjacent* to  $v$ ; its cardinality  $|N_G(v)|$  is also called the *degree* of  $v$ , denoted as  $\deg_G(v)$ . We denote the *closed neighborhood* of  $v$  by  $N_G[v] = N_G(v) \cup \{v\}$ . We can extend set-valued functions to set arguments; for instance,  $N_G[S] = \bigcup_{v \in S} N_G[v]$  for a set of vertices  $S$ ;  $S$  is a *dominating set* if  $N_G[S] = V$ . Whenever clear from context, we may drop the subscript  $G$  from our notation. If  $X \subseteq V$ , we also write  $N_X(v)$  instead of  $N(v) \cap X$  and  $\deg_X(v)$  for  $|N(v) \cap X|$ . For brevity, we write CDS for *connected dominating set*. Next, we collect some observations.

► **Observation 1.** *If  $S$  is a CDS of a partial graph  $G'$  of  $G$ , then  $S$  is a CDS of  $G$ .*

**Proof.** We can think of  $G = (V, E)$  as being obtained from  $G'$  by adding edges. Hence, if  $N_{G'}[S] = V$ , then  $N_G[S] = V$ . Moreover, adding edges cannot violate connectivity. ◀

► **Corollary 2.1.** *Let  $S$  be a CDS both of  $G$  and of a partial graph  $G'$  of  $G$ . If  $S$  is a minimal CDS of  $G$ , then it is a minimal CDS of  $G'$ .*

A graph  $G = (V, E)$  is *d-degenerate* if there exists an *elimination ordering*  $(v_1, \dots, v_n)$ , where  $V = \{v_1, \dots, v_n\}$ , such that, for all  $i = 1, \dots, n$ ,  $\deg_{G[\{v_i, \dots, v_n\}]}(v_i) \leq d$ . In other words, we can subsequently delete  $v_1, v_2, \dots$  from  $G$ , and at the time when  $v_i$  is deleted, it has degree bounded by  $d$  in the remaining graph. The decision problem CONNECTED DOMINATING SET EXTENSION expects as inputs a graph  $G = (V, E)$  and a vertex set  $U$ , and the question is if there exists a minimal CDS  $S$  that extends  $U$ , *i.e.*, for which  $S \supseteq U$  holds.

In the next section, we develop a branching algorithm. It is classical to analyze its running-time by solving recurrences of type  $T(\mu(\mathcal{I})) = \sum_{i=1}^t T(\mu(\mathcal{I}) - r_i)$ . Here,  $\mu(\mathcal{I})$  is a measure on the size of the instance. The value  $t$  is the number of recursive calls ( $t$  is equal to 1 for *reduction rules*) and each  $r_i$  is (a lower bound on) the reduction of the measure corresponding to the recursive call. We simply denote by  $(r_1, r_2, \dots, r_t)$  the *branching vector* of the recurrence. We refer to the book by Fomin and Kratsch for further details on this standard analysis [17].

As discussed in the introduction, we shall first prove that all minimal CDS can be enumerated in time  $\mathcal{O}(1.9767^n)$  on 2-degenerate graphs. This result is the key to our enumeration result for general graphs. This is why we will first present the corresponding branching enumeration algorithm for 2-degenerate graphs in a simplified form and analyze it with a rather simple measure in order to explain its main ingredients, and only thereafter, we turn towards a refined analysis that finally leads to the claimed enumeration result on general graphs. We shall prove that our input-sensitive enumeration algorithm cannot be turned into an enumeration algorithm with polynomial delay by simply interleaving the branching with tests for extendibility. For possible applications of such extension algorithms, we refer to the discussions in [7].

### 3 A CDS enumeration algorithm for 2-degenerate graphs

We are going to present an algorithm that enumerates all inclusion-wise minimal connected dominating sets (CDS) of a 2-degenerate graph  $G = (V, E)$ . Based on  $G$ , in the course of our algorithm, an instance is specified by  $\mathcal{I} = (V'; O_d, O_n; S)$ , consisting of four vertex sets that partition  $V$ . In the beginning,  $\mathcal{I} = (V; \emptyset; \emptyset; \emptyset)$ . In general,  $V'$  collects the vertices not yet decided by branching or reduction rules,  $O := O_d \cup O_n$  are the vertices that have been decided *not* to be put into the solution, while  $S$  is the set of vertices decided to go into the solution that is constructed by the branching algorithm. The set  $O$  is further refined into  $O_n$ , the set of vertices that are not yet dominated, *i.e.*, if  $x \in O_n$ , then  $\deg_S(x) = 0$ , and  $O_d$ , the set of vertices that are already dominated, *i.e.*, if  $x \in O_d$ , then  $\deg_S(x) > 0$ . Similarly, we will sometimes refine  $V' = V'_d \cup V'_n$ . Notice that also vertices known to be dominated could be still put in the dominating set, either to dominate other vertices or for connectivity purposes. In the leaves of the branching tree, only instances of the form  $(\emptyset, O_d, \emptyset, S)$  are of interest. Yet, before outputting  $S$  as a solution, one has to check if  $S$  is connected and if it does not contain a smaller CDS.

The algorithm actually starts by creating  $n$  different branches, in each a single vertex is put in  $S$  as a starting point, so that  $S$  is never empty. More precisely, the  $n^{\text{th}}$  branch would decide *not* to put the previously considered  $n - 1$  vertices into the solution but the  $n^{\text{th}}$  one is put into  $S$ . This binary branching avoids generating solutions twice. Also, it is trivial to check in each branch if the selected vertex already dominates the whole graph, so that we can henceforth assume that  $G$  cannot be dominated by a single vertex.

We denote by  $c$  the number of connected components of  $G[S]$ . Now, we are ready to define the measure that we use to analyze the running time of our algorithm, following a very simple version of the *measure-and-conquer*-paradigm, as explained in [15, 17],

$$\mu(\mathcal{I}) = |V'| + \alpha \cdot |O_n| + \delta \cdot c.$$

We decide that  $0 < \alpha, \delta < 1$ , but we will determine the concrete values later as to minimize the upper-bound on the running-time. At the beginning,  $V' = V$ ,  $O_n = \emptyset$  and  $c = 0$ , so that then the measure equals  $|V|$ . At the end,  $V' = O_n = \emptyset$  and the measure equals  $\delta$  if the solution is connected and is bigger than  $\delta$  if the solution is not connected.

For the possible branchings, we only consider vertices in a partial graph  $G'$  of  $G[V' \cup O_n]$ . As  $G$  is 2-degenerate,  $G'$  is also 2-degenerate, so that we can always find a vertex of degree at most two in  $G'$ . Some of our branchings apply to vertices of arbitrary degree, though; in such a situation, we denote the vertex that we branch on as  $x$ . If we branch on a small-degree vertex (due to 2-degeneracy), this vertex is called  $u$ . Clearly, a binary branch that puts a selected vertex either into  $S$  or into  $O$  is a complete case distinction.



(a)  $x \in V'_d$  has neighbors in  $O_n$ .

(b) Vertices from two different connected components  $S_1, S_2$  of  $G[S]$  dominate  $x$  in  $G$ .

■ **Figure 1** Simple branchings for dominated vertices: Rules 1 and 2.

We are now explaining the conventions that we follow in our illustrations of subgraphs of  $G'$ . Vertices in  $V'$  are depicted by  $\ominus$  and more specifically by  $\circ$  if in  $V'_n$  or by  $\bullet$  if in  $V'_d$ . We use black squares  $\blacksquare$  to depict vertices which are already decided to belong to the solution  $S$ . We use  $\square$  for vertices from  $O_n$ . So, circles are used for undecided vertices (these vertices might still be added to  $S$ ), whereas squares are used for vertices being already decided (to belong to the solution or to be discarded). Vertices in  $V' \cup O$  are depicted as half-filled diamonds  $\blacklozenge$ , and if the vertex is from  $V'_n \cup O_n$ , then we use an unfilled diamond  $\diamond$  to represent it. A dashed line indicates an edge that may be present. It should be clear that one could always move to the graph where vertices in  $S$  that belong to the same connected component in  $G[S]$  are merged. In order to avoid drawing too many vertices from  $S$  in our pictures, we assume these mergings to have been performed. Hence, when we draw two vertices from  $S$ , they belong to different connected components.

In the following, the branching and reduction rules require to be executed in order, so that our instance will (automatically) satisfy some structural properties when we apply one of the later rules. We can separate our branching and reduction rules into three parts:

- A first set of rules (Branching Rules 1, 2, 3, 4, Reduction Rules 1, 2, 3, 5) that deals with vertices of arbitrary degree that are (possibly) dominated, but only in some special cases, as detailed later. Those rules are applied first, so that whenever we apply a rule from the next two sets, we know that any dominated vertex is dominated by vertices from exactly one connected component of  $G[S]$  and none of the vertices in its neighborhood are dominated. This means that the set of vertices  $V'_d$  as well as the set of vertices  $O_n$  forms an independent set in  $G'$ , a partial graph of  $G[V' \cup O_n]$ .
- A second set of rules (Branching Rules 5, 6, Reduction Rule 4) that handles the cases where the small-degree vertex that exists by the 2-degeneracy is undominated. If we apply a rule from the third set, every undominated vertex is of degree at least 3.
- The last set of rules (Branching Rules 7, 8, 9, 10) handles only the cases where the small-degree vertex that exists by the 2-degeneracy is dominated.

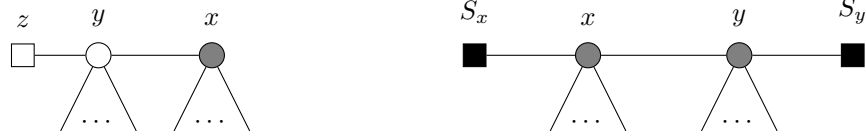
Note that even inside those three sets, rules have to be executed in the given order.

► **Branching Rule 1.** Let  $x \in V'_d$  with  $\deg_{O_n}(x) \geq 1$  (Figure 1a). Then branch as follows.

- (1) Put  $x$  in  $O_d$ .
- (2) Put  $x$  in  $S$  and every vertex in  $N_{O_n}(x)$  in  $O_d$ .

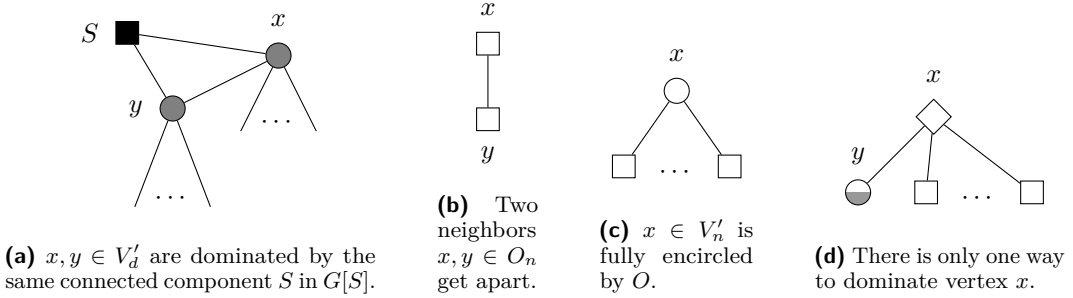
► **Lemma 3.1.** *The branching of rule 1 is a complete case distinction. Moreover, it leads to a branching vector that is not worse than  $(1, 1 + \alpha)$ .*

**Proof.** As  $\deg_S(x) \geq 1$ , we find that, if  $x$  is not put into the solution, then it is put into  $O_d$ , which decreases the measure by 1 in the first component of the branching vector. If  $x$  is put into the solution, then its neighbors are dominated and we decrease the measure by at least  $1 + \alpha$  in the second component of the branching vector, as  $\deg_{O_n}(x) \geq 1$ . ◀



(a) A vertex from  $O_n$  in the second neighborhood of  $x \in V'_d$  gives still an advantage. (b)  $N_S(x)$  belongs to the same connected component  $S_x$  of  $G[S]$ , and so does  $N_S(y)$  belong to  $S_y$ , but  $S_x \neq S_y$ .

■ Figure 2 Branching Rules 3 and 4.



(a)  $x, y \in V'_d$  are dominated by the same connected component  $S$  in  $G[S]$ . (b) Two neighbors  $x, y \in O_n$  get apart. (c)  $x \in V'_n$  is fully encircled by  $O$ . (d) There is only one way to dominate vertex  $x$ .

■ Figure 3 Illustrating Reduction Rules 1, 3, 4 and 5.

We will have similar lemmas for each of the branching rules; we will summarize them at the end of this section instead of formulating them separately and point to the long version of the paper.

► **Branching Rule 2.** Let  $x \in V'_d$  such that  $x$  is adjacent to two different connected components of  $G[S]$ ; see Figure 1b. Then branch as follows. (1) Put  $x$  in  $O_d$ . (2) Put  $x$  in  $S$ .

We are now presenting two branching rules that could be viewed as variations of the first two; they always give worse branchings.

► **Branching Rule 3.** Let  $x \in V'_d, y \in N_{V'_n}(x), z \in N_{O_n}(y)$  (Figure 2a). Then branch as follows. (1) Put  $x$  in  $O_d$ . (2) Put  $x$  in  $S, y$  in  $O_d$ . (3) Put  $x, y$  in  $S$  and thus  $z \in O_d$ .

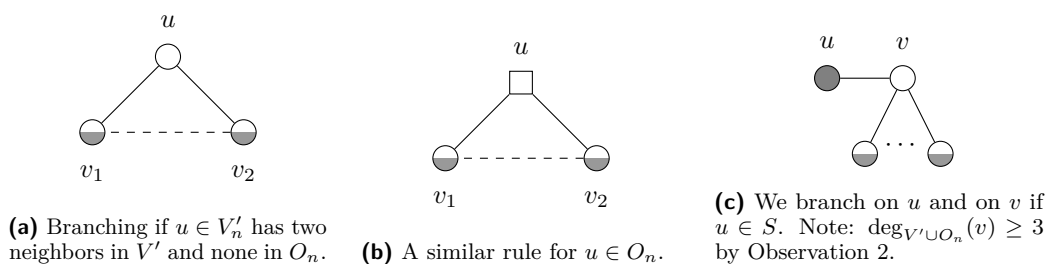
► **Branching Rule 4.** Let  $x, y \in V'_d, xy \in E$ , such that  $z \in \{x, y\}$  is adjacent to a connected component  $S_z$  of  $G[S]$ , with  $S_x \neq S_y$ , see Figure 2b. Then branch as follows. (1) Put  $x$  in  $O_d$ . (2) Put  $x$  in  $S$  and  $y$  in  $O_d$ . (3) Put  $x$  in  $S$  and  $y$  in  $S$ .

Notice that in the last branch, the number of connected components decreases.

► **Reduction Rule 1.** If  $x, y \in V'_d$  and  $xy \in E$ , then delete the edge  $xy$ ; see Figure 3a.

► **Lemma 3.2.** *Reduction Rule 1 is sound and the measure does not change.*

**Proof.** Let  $M$  be a minimal CDS of  $G$  such that  $M \setminus V' = S$ . Define  $e = xy$  and  $\tilde{G} = (V, E \setminus \{e\})$ . Now we want to show that  $M$  is also a minimal CDS in  $\tilde{G}$ . Since  $x$  and  $y$  are already dominated by  $S$ , the deletion of the edge  $e$  would not affect domination, nor could  $x$  ever be the private neighbor of  $y$  or vice versa. The connectivity is only important if  $x, y \in M$ . Vertices  $x, y$  are dominated by the same connected component of  $S$ , as otherwise Branching Rule 4 would have applied with priority. Hence, there exists a path  $p = (x, p_1, \dots, p_l, y)$ , with internal vertices in  $S$ . Let  $q = (q_1, \dots, q_k)$  be a path in  $G[M]$  such that there exists an  $i \in \{1, \dots, k-1\}$  with  $q_i = x$  and  $q_{i+1} = y$ . Then  $\tilde{q} = (q_1, \dots, q_i, p_1, \dots, p_l, q_{i+1}, \dots, q_k)$  is a walk in  $\tilde{G}[M]$ . Thus,  $\tilde{G}[M]$  is connected and  $M$  is a CDS of  $\tilde{G}$ . As  $\tilde{G}$  is a partial graph,  $M$  is also a minimal CDS of  $G$  by Corollary 2.1. ◀



■ **Figure 4** Branching Rules 5, 6 and 7.

We can formulate and prove similar lemmas for the following reduction rules, as well. We will summarize this in one single lemma below instead.

- ▶ **Reduction Rule 2.** If  $x$  is an isolated vertex in  $G[V' \cup O_n]$ , do the following:
  - If  $x$  is dominated, put  $x$  into  $O_d$ .
  - If  $x$  is not dominated, skip this branch. (This will always happen if  $x \in O_n$ .)
- ▶ **Reduction Rule 3.** Let  $x, y \in O_n$  be with  $xy \in E$ ; see Figure 3b. We delete the edge  $xy$ .
- ▶ **Reduction Rule 4.** If  $x \in V'_n$  obeys  $N(x) \cap V' = \emptyset$ , then skip this branch (Figure 3c).

The next rule will even decrease the measure by at least  $1 - \delta$ .

- ▶ **Reduction Rule 5.** Let  $x \in V'_n \cup O_n$ , with  $N_{V'}(x) = \{y\}$ . Then, put  $y$  into  $S$ .
- ▶ **Branching Rule 5.** Let  $u \in V'_n$  with  $\deg_{V'}(u) = 2$ ,  $\deg_{O_n}(u) = 0$  and  $N_{V'}(u) = \{v_1, v_2\}$ ; see Figure 4a. Then branch as follows. (1) Put  $u$  in  $O_d$ ,  $v_1$  in  $S$ . (2) Put  $u$  in  $O_d$ ,  $v_1$  in  $O$ ,  $v_2$  in  $S$ . (3) Put  $u$  in  $S$ ,  $v_1$  in  $S$ . (4) Put  $u$  in  $S$ ,  $v_1$  in  $O_d$ ,  $v_2$  in  $S$ .

More precisely, in the second branch, we put  $v_1$  into  $O_d$  if  $v_1v_2 \in E$  or if  $v_1$  was already dominated and we put  $v_1$  into  $O_n$ , otherwise. The same is done in the Branching Rules 6, 7, 8, 9 and 10, as it can be decided whether the vertex goes into  $O_n$  or into  $O_d$ .

- ▶ **Branching Rule 6.** Let  $u \in O_n$  with  $N_{V'}(u) = \{v_1, v_2\}$  (Figure 4a). Then branch as follows. (1) Put  $v_1$  in  $S$ , and thus  $u$  in  $O_d$ . (2) Put  $v_1$  in  $O$ ,  $v_2$  in  $S$ , and thus  $u$  in  $O_d$ .

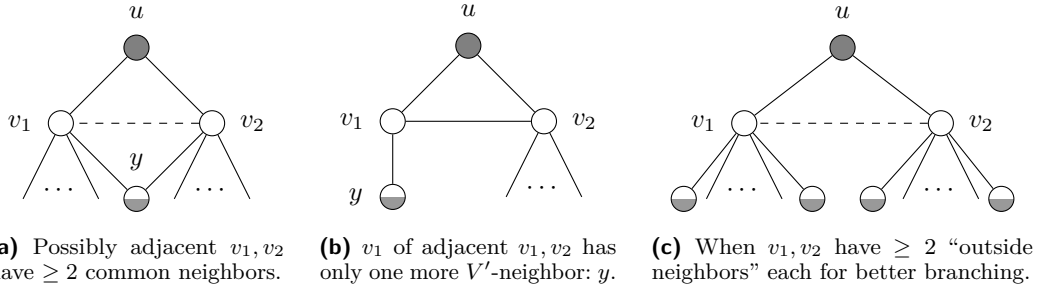
▶ **Observation 2.** For each rule below, as it is applied in particular after Branching Rule 5 and Reduction Rule 5, we observe: for any vertex  $v \in V'_n \cup O_n$ , we have  $\deg_{V' \cup O_n}(v) \geq 3$ .

- ▶ **Branching Rule 7.** Let  $u \in V'_d$  with  $N_{V'}(u) = \{v\}$  and  $\deg_{O_n}(v) = 0$ ; see Figure 4c. Then branch as follows. (1) Put  $u$  in  $O_d$ . (2) Put  $u$  in  $S$  and  $v$  in  $O_d$  and thus all the vertices of  $N_{V'}(v) \setminus \{u\}$  in  $O$ . (3) Put  $u$  in  $S$  and  $v$  in  $S$ .

- ▶ **Branching Rule 8.** Let  $u \in V'_d$ , with  $N_{V'}(u) = \{v_1, v_2\}$  and  $\deg_{O_n}(u) = 0$  (Figure 5a). If  $N_{V'}(v_1) \cap N_{V'}(v_2)$  contains a vertex  $y \in V'$  different from  $u$ , then branch as follows. (1) Put  $u$  in  $O_d$ . (2) Put  $u$  in  $S$ ,  $v_1$  in  $S$ . (3) Put  $u$  in  $S$ ,  $v_1$  in  $O_d$ ,  $v_2$  in  $S$ . (4) Put  $u$  in  $S$ ,  $v_1$  in  $O_d$ ,  $v_2$  in  $O_d$  and thus  $y$  in  $O$ .

- ▶ **Branching Rule 9.** Let  $u \in V'_d$ ,  $N_{V'}(u) = \{v_1, v_2\}$ ,  $N_{V'_d}(u) = N_{O_n}(u) = \emptyset$  and  $N_{V'}(v_1) \setminus \{u, v_2\} = \{y\}$ ; see Figure 5b. Then branch as follows. (1) Put  $u$  in  $O_d$ . (2) Put  $u$  in  $S$ ,  $v_1$  in  $S$ . (3) Put  $u$  in  $S$ ,  $v_1$  in  $O_d$ ,  $v_2$  in  $S$ . (4) Put  $u$  in  $S$ ,  $v_1$  in  $O_d$ ,  $v_2$  in  $O_d$  and  $y$  in  $O$ . (5) Put  $u$  in  $S$ ,  $v_1$  in  $O_d$ ,  $v_2$  in  $O_d$  and  $y$  in  $S$  and thus the vertices of  $N_{V'}(v_2) \setminus \{u, v_1\}$  in  $O$ .

- ▶ **Branching Rule 10.** (Figure 5c) Let  $u \in V'_d$ ,  $N_{V'}(u) = \{v_1, v_2\}$ ,  $N_{V'_d}(u) = N_{O_n}(u) = \emptyset$ , such that  $|N_{V'}(v_1) \setminus \{u, v_2\}| \geq 2$ , as well as  $|N_{V'}(v_2) \setminus \{u, v_1\}| \geq 2$ . Then branch as follows. (1) Put  $u$  in  $O_d$ . (2) Put  $u$  in  $S$ ,  $v_1$  in  $S$ . (3) Put  $u$  in  $S$ ,  $v_1$  in  $O_d$ ,  $v_2$  in  $S$ . (4) Put  $u$  in  $S$ ,  $v_1$  in  $O_d$ ,  $v_2$  in  $O_d$  and all of  $N_{V'}(v_1) \setminus \{u, v_2\}$  in  $O$ . (5) Put  $u$  in  $S$ ,  $v_1$  in  $O_d$ ,  $v_2$  in  $O_d$  and all of  $N_{V'}(v_2) \setminus \{u, v_1\}$  in  $O$ .



■ **Figure 5** Branching Rules 8, 9 and 10.

The next and final rule will never be applied when this algorithm is applied to a 2-degenerate graph. It rather prepares the ground for the general case. However, with our current measure, this would yield an  $\mathcal{O}(2^n)$  algorithm in the general case. With a modified measure, as described in the next section, we will achieve a better running time.

► **Branching Rule 11.** Let  $x \in V'_d$ . Then we branch as follows. (1) Put  $x$  in  $O_d$ . (2) Put  $x$  in  $S$  and thus the vertices of  $N_{V'}(x)$  in  $V'_d$ .

► **Lemma 3.3.** *Each of the mentioned branching rules covers all cases in their described respective situation. The branching will lead to a branching vector as listed in Table 1.*

► **Lemma 3.4.** *The mentioned reduction rules are sound and their application never increases the measure.*

A case analysis shows the correctness of our algorithm in the following sense:

► **Lemma 3.5.** *The Reduction and Branching Rules cover all possible cases.*

**Proof.** As explained below, Branching Rule 11 serves as a final catch-all. For the 2-degenerate case, we should focus on all other rules and prove that they cover all cases. This means that we have to show that the proposed algorithm will resolve each 2-degenerate graph completely. Our rule priorities might remove vertices of arbitrary degree from the graph  $G'$  that is a partial graph of the graph  $G[V' \cup O_n]$ . This way, we again arrive at a 2-degenerate graph. Yet, what is important for dealing with 2-degenerate graphs is to consider all cases of a vertex  $u$  of degree at most 2. The degree conditions in the following case distinction refer to  $G'$ . Degree-0 vertices are treated with Reduction Rule 2. We now discuss vertices  $u$  of degrees one or two. There are two different cases: either  $u$  is in  $V'_n \cup O_n$ , or  $u \in V'_d$ .

**Case 1:**  $u$  is not dominated by  $S$ . Now we discuss its degree in  $G'$ , either it is 1 or it is 2.

**Case 1.1:**  $\deg(u) = 1$  and we denote by  $v$  the neighbor in  $G'$  of  $u$ . If  $v \in O_n$ , then either  $u$  is in  $O_n$  and then satisfies the conditions of Reduction Rule 3, or it is in  $V_n$  and thus satisfies the conditions of Reduction Rule 4. If  $v \notin O_n$ ,  $v$  satisfies the conditions of Reduction Rule 5.

**Case 1.2:**  $\deg(u) = 2$ . Now we discuss the number of neighbors of  $u$  in  $O_n$ . If both neighbors are in  $O_n$ , Reduction Rule 4 applies if  $u \in V'_n$  and Reduction Rule 3 applies otherwise. If only one neighbor of  $u$  is in  $O_n$ , then this means that if  $u \in O_n$  Reduction Rule 3 applies, and otherwise Reduction Rule 5 applies. If none of the neighbors are in  $O_n$ , then either  $u \in V'_n$  and then Branching Rule 5 applies, or otherwise  $u \in O_n$  and thus Branching Rule 6 applies.

**Case 2:**  $u$  is dominated by  $S$ .



■ **Table 1** Collection of all branching vectors for the 2-degenerate case; the branching numbers are displayed for the different cases with  $\alpha = 0.106$  and  $\delta = 0.106$ .

Branching Rule #	Branching vector	Branching number
1	$(1, 1 + \alpha)$	always better than 3
2	$(1, 1 + \delta)$	always better than 4
3	$(1, 2, 2 + \alpha)$	$< 1.9766$
4	$(1, 2, 2 + \delta)$	$< 1.9766$
5	$(2 - \delta, 3 - \delta - \alpha, 2 - \delta, 3 - \delta)$	$< 1.8269$
6	$(1 + \alpha - \delta, 2 - \delta)$	$< 1.6420$
7	$(1, 4 - 2\alpha, 2)$	$< 1.7691$
8	$(1, 2, 3, 4 - \alpha)$	$< 1.9333$
9	$(1, 2, 3, 4 - \alpha, 5 - \delta - \alpha)$	$< 1.9767$
10	$(1, 2, 3, 5 - 2\alpha, 5 - 2\alpha)$	$< 1.9420$
11	$(\beta, 3 - 2\beta)$ where $\beta = 1$	$= 2$ not for 2-degenerate graphs

**Case 2.1:**  $\deg(u) = 1$ . We denote by  $v$  the neighbor in  $G'$  of  $u$ . If  $v$  is dominated by  $S$  then Reduction Rule 1 applies. If  $v$  is not dominated by  $S$ , then either it has no neighbors in  $O_n$  and then Branching Rule 7 applies, or it has at least one neighbor in  $O_n$  and Branching Rule 3 applies.

**Case 2.2:**  $\deg(u) = 2$ . We denote  $\{v_1, v_2\} = N_{V'}(u)$ . If either  $v_1$  or  $v_2$  is dominated, then Reduction Rule 1 applies. If both of them are not dominated by  $S$ , either they have at least one neighbor in  $O_n$  and then Branching Rule 3 applies or they do not have any neighbor in  $O_n$ . So we know that  $\deg_{V'}(v_1) \geq 3$ , and  $\deg_{V'}(v_2) \geq 3$ , otherwise we could apply one of the rules of the previous case to  $v_1$  or  $v_2$ . Now, either  $v_1$  and  $v_2$  have a common neighbor outside of  $u$  and Branching Rule 8 applies, or they do not and then either at least one of them has exactly one neighbor that is not  $u$  or  $v_1$  (or  $v_2$ , respectively) and Branching Rule 9 applies, or they both have at least two neighbors outside of  $u, v_1$  and  $v_2$ , so that Branching Rule 10 applies.

We finally have to prove the correctness of the algorithm for a general graph. If any vertex satisfies the conditions of any Branching or Reduction Rule apart from Branching Rule 11, then we apply such a rule, otherwise no such rule applies, which means that the minimum degree of  $G'[V' \cup O_n]$  is 3. If at least one vertex is in  $V_d$ , then we can apply Branching Rule 11. Assume it is not the case, that means  $V_d$  is empty, so every vertex of  $V' \cup O_n$  is not dominated (and it is not empty, otherwise the algorithm would have ended). Since in the beginning,  $S$  was not empty,  $S$  is never empty. Moreover, at any point in the algorithm,  $N(S) = V_d \cup O_d$ . So in our case, we have  $N(S) = O_d$ . This means that  $S$  is not a dominating set (there are vertices in  $V_n \cup O_n$ ) and  $S$  cannot have any neighbor added, so there is no CDS  $M \subset V$  such that  $M \setminus V' = S$ . This branch has to be discarded. ◀

► **Theorem 3.6.** *CONNECTED DOMINATING SET ENUMERATION can be solved in time  $\mathcal{O}(1.9767^n)$  on 2-degenerate graphs of order  $n$ , using polynomial space only. The claimed branching number is attained by setting  $\alpha = 0.106$  and  $\delta = 0.106$ ; see Table 1.*

► **Corollary 3.7.** *2-degenerate graphs have no more than  $\mathcal{O}(1.9767^n)$  minimal CDS.*

► **Remark 3.8.** We could deduce a corresponding CDS enumeration result for subcubic graphs, as they can be dealt with as 2-degenerate graphs after the initial branching. There is a clear indication that the bound that we derive in this way is not tight. Namely, Kangas et al. have shown in [25] that there are no more than  $1.9351^n$  many connected sets of vertices in a subcubic graph of order  $n$ .

**4 Getting  $\beta$  into the game: the general case**

As indicated above, we are now refining the previous analysis by splitting the set of vertices of the currently considered graph further. More precisely, the set of vertices  $V'$  that have not been decided to come into or to be out of the solution  $S$  is split into the set of vertices  $V'_n$  that is still undominated and the set  $V'_d$  of vertices that is already dominated. From the viewpoint of the original graph, the neighbors of the solution set  $S$  are therefore partitioned into the sets  $V'_d$  and  $O_d$ . However, observe that although we do not consider  $O_d$  anymore in the present graph, we do care about  $V'_d$ , since  $V'_d$ -vertices can still be either placed into  $S$  or into  $O_d$  by future branching or reduction rules. This is also reflected in the measure of the instance  $I$ , which is now defined as:

$$\mu(I) = |V'_n| + \alpha \cdot |O_n| + \beta \cdot |V'_d| + \delta \cdot c$$

We set  $\alpha = 0.110901$ ,  $\beta = 0.984405$  and  $\delta = 0.143516$ . We are next working through the branching rules one by one, as in particular the branching vectors will now split off into different cases, as in our preliminary analysis, we only took care of the worst cases, not differentiating between  $V'_n$  or  $V'_d$  (which was summarized under the set name  $V'$  before). For the convenience of the reader, we repeat the formulation of the branching rules in the following, adapting the notations.

We start with a table stating the branching vectors according to this new measure for some branching rules for which it is straightforward. Branching Rule 3 and Branching Rule 4 are the tight cases in our Measure-and-Conquer analysis.

Rule	Branching vector	Br. number	Rule	Branching vector	Br. number
# 1	$(\beta, \beta + \alpha)$	< 1.9489	# 2	$(\beta, \beta + \delta)$	< 1.9297
# 3	$(\beta, \beta + 1, \beta + 1 + \alpha)$	< 1.9896	# 4	$(\beta, 2\beta, 2\beta + \delta)$	< 1.9896

Branching Rule 5 Let  $u \in V'_n$  with  $\deg_{V'}(u) = 2$ ,  $\deg_{O_n}(u) = 0$  and  $N_{V'}(u) = \{v_1, v_2\}$ . The branching vector is different when  $v_1, v_2 \in V'_d$  or  $v_1, v_2 \in V'_n$ . We will assume  $v_1 v_2 \notin E$ , as this is always the worst case. A similar analysis applies to Branching Rule 6.

Condition	Branching vector for # 5	Br. number	Br. vector for # 6	Br. number
$v_1, v_2 \in V'_n$	$(2 - \delta, 3 - \delta - \alpha, 2 - \delta, 3 - \delta)$	< 1.8463	$(1 + \alpha - \delta, 2 - \delta)$	< 1.6635
$v_1 \in V'_n, v_2 \in V'_d$	$(2 - \delta, 2 - \alpha + \beta, 2 - \delta, 2 + \beta)$	< 1.8236	$(1 + \alpha, 1 + \beta)$	< 1.5855
$v_1, v_2 \in V'_d$	$(1 + \beta, 1 + 2\beta, 1 + \beta, 1 + 2\beta)$	< 1.7785	$(\beta + \alpha, 2\beta + \alpha)$	< 1.5817

Branching Rule 7 We look at  $u \in V'_d$ ,  $\{v\} = V'_n \cap N(u)$  and the neighbors of  $v$ ; we know that  $N(v)$  contains at least  $u, v_1, v_2 \notin O$ ; differentiating between  $v_i \in V'_n$  or  $v_i \in V'_d$  (as before), we arrive at three cases never leading to a branching number worse than 1.78.

Branching Rule 8 Let  $u \in V'_d$  be with  $\deg_{V' \cup O_n}(u) = 2$ , such that  $v_1, v_2 \in N_{V'}(u)$  and  $N_{V'}(v_1) \cap N_{V'}(v_2)$  contains a vertex  $y \in V'$  different from  $u$ . We differentiate  $y \in V'_n$  or  $y \in V'_d$  and arrive at a branching number not worse than 1.9403.

Branching Rule 9 Let  $u \in V'_d$  be with  $\deg_{V' \cup O_n}(u) = 2$ , such that  $v_1, v_2 \in N_{V'}(u)$  and  $N_{V'}(v_1) \setminus \{u, v_2\} = \{y\}$ . We distinguish  $y \in V'_n$  and  $y \in V'_d$ ; the first case leads to another tight-case branching for our algorithm. In the last branch of each vector,  $\min(\beta, 1 - \alpha)$  corresponds to whether  $z \in N(v_2) \setminus \{u, v_1\}$  belongs to  $V'_d$  or to  $V'_n$ .

Condition	Branching vector	Branching number
$y \in V'_n$	$(\beta, 1 + \beta, 2 + \beta, 3 + \beta - \alpha, 3 + \beta - \delta + \min(\beta, 1 - \alpha))$	< 1.9896
$y \in V'_d$	$(\beta, 1 + \beta, 2 + \beta, 2 + 2\beta, 2 + 2\beta + \min(\beta, 1 - \alpha))$	< 1.9813

**Branching Rule 10** Let  $u \in V'_d$ ,  $N_{V'_n}(u) = \{v_1, v_2\}$ ,  $N_{V'_d}(u) = N_{O_n}(u) = \emptyset$ . We further assume that  $|N_{V'}(v_1) \setminus \{u, v_2\}| \geq 2$ , as well as  $|N_{V'}(v_2) \setminus \{u, v_1\}| \geq 2$ . The only thing to discuss here is whether the vertices of  $N_{V'}(v_1) \setminus \{u, v_2\}$ ,  $N_{V'}(v_2) \setminus \{u, v_1\}$  are in  $V'_n$  or  $V'_d$ . This leads to nine subcases, which never produce a branching vector worse than 1.9453.

**Branching Rule 11** When this branching rule applies, the sets in which the different vertices reside are all already known: as it is applied after Branching Rule 1 and Branching Rule 4, all the neighbors of  $x \in V'_d$  are in  $V'_n$ , moreover,  $\deg_{V'_n}(x) = \deg_{V' \cup O_n}(x) \geq 3$ , as it is applied after every other rule. The branching vector  $(\beta, 3 - 2\beta)$  gives a branching number  $< 1.9896$ , which is the last tight-case branching.

► **Theorem 4.1.** *CONNECTED DOMINATING SET ENUMERATION can be solved in time  $\mathcal{O}(1.9896^n)$  on graphs of order  $n$ , using polynomial space only.*

► **Corollary 4.2.** *There are no more than  $\mathcal{O}(1.9896^n)$  many minimal connected dominating sets in a graph of order  $n$ .*

## 5 Achieving polynomial delay is not easy

How could we achieve polynomial delay for enumeration problems? If CONNECTED DOMINATING SET EXTENSION would be solvable in polynomial time, then we might cut search tree branches whenever it is clear that no solution is to be expected beyond a certain node of the search tree, as we can associate to such a node also a set of vertices  $U$  that is decided to go into the solution. For example, it has been recently exemplified with the enumeration problem of minimal Roman dominating functions in [2] that a polynomial-time algorithm for the corresponding extension problem can be adapted so that polynomial delay can be achieved for this type of enumeration problem. However, we can show that CONNECTED DOMINATING SET EXTENSION is NP-complete by a reduction from 3-SAT. This means that new ideas would be needed for showing polynomial delay for enumerating minimal connected dominating sets.

► **Theorem 5.1.** *The CONNECTED DOMINATING SET EXTENSION problem is NP-complete, even when restricted to 2-degenerate graphs.*

Due to the parsimonious nature of the reduction, we can also conclude the following result.

► **Corollary 5.2.** *Assuming that the Exponential Time Hypothesis<sup>1</sup> holds, there is no algorithm that solves the CONNECTED DOMINATING SET EXTENSION problem in time  $\mathcal{O}(2^{o(n)})$  on (2-degenerate) graphs of order  $n$ .*

Hence, even any subexponential delay seems to be hard to achieve.

Furthermore, the CONNECTED DOMINATING SET EXTENSION with the *standard parameterization*  $|U|$ , where  $U$  is the given subset of  $V$ , is even W[3]-hard on split graphs. To show this, we need the W[3]-completeness of HITTING SET EXTENSION [4, 5, 7] with standard parameterization; in this problem, the input consists of a hypergraph  $(X, \mathcal{S})$ , with  $\mathcal{S} \subseteq 2^X$ , and a set  $U \subseteq X$ , and the question if there exists a minimal hitting set  $H$  (this means that  $H \cap S \neq \emptyset$  for all  $S \in \mathcal{S}$ ) that extends  $U$ , i.e.,  $U \subseteq H$ .

► **Theorem 5.3.** *CONNECTED DOMINATING SET EXTENSION (with standard parameterization) is W[3]-hard, even on split graphs.*

<sup>1</sup> For a discussion of this hypothesis, we refer to [23, 28].

On split graphs, we can also prove W[3]-completeness of CONNECTED DOMINATING SET EXTENSION. Few natural parameterized problems are known to be W[3]-complete [4, 5, 7, 8]. There is yet another stroke against hopes for obtaining a polynomial-delay enumeration algorithm for minimal CDS. Namely, we could see the reduction presented for Theorem 5.3 also as a reduction that proves the following statement, which connects our enumeration problem to HITTING SET TRANSVERSAL, a problem notoriously open for decades.

► **Theorem 5.4.** *If there was an algorithm for enumerating minimal CDS with polynomial delay in split graphs, then there would be an algorithm for enumerating minimal hitting sets in hypergraphs with polynomial delay.*

## 6 Lower bounds

Several attempts to construct lower bound examples are known from the literature, leading to  $3^{(n-2)/3} \in \Omega(1.4422^n)$  many minimal connected dominating sets in  $n$ -vertex graphs [20, 33]. We now present an improved lower bound on the maximum number of minimal connected dominating sets in a graph. Given arbitrary positive integers  $k, t$ , we construct a graph  $G_t^k$  of order  $n = k(2t + 1) + 1$  as follows. The main building blocks of  $G_t^k$  consist of  $k$  copies of a base-graph  $G_t$ , of order  $2t - 1$ . The vertex set of  $G_t$  consists of three layers. The first one is a set  $X = \{x_1, \dots, x_t\}$  that induces a clique. The second one is an independent set  $Y = \{y_1, \dots, y_t\}$ , while the third layer consists of a singleton  $\{z\}$ . Each vertex  $x_i \in X$  has exactly  $t - 1$  neighbors in  $Y$ :  $N(x_i) = \{y_j \in Y : i \neq j\}$ . Hence,  $X \cup Y$  induces a copy of  $K_{t,t}$  minus a perfect matching. Finally the vertex  $z$  is adjacent to all vertices in  $Y$ . Figure 6a shows the graph  $G_t$  for  $t = 4$ . To finally construct the graph  $G_t^k$ , we introduce a final vertex  $s$  that is connected to all vertices of each set  $X$  of each copy of the base-graph.

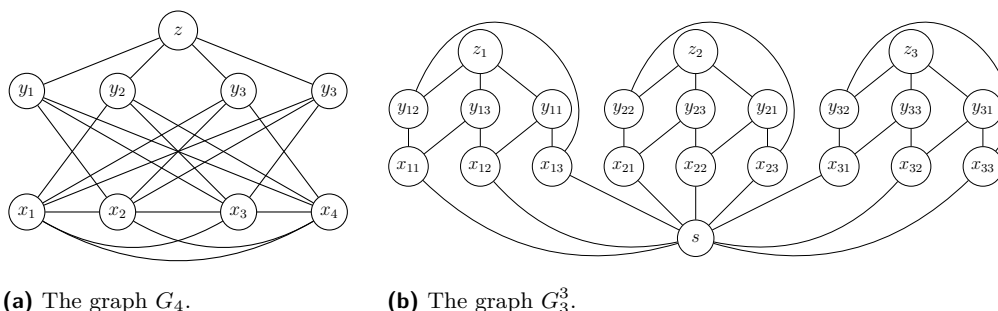
► **Lemma 6.1.** *For each  $t > 0$ , the graph  $G_t$  has exactly  $\frac{t^3+t^2}{2} - t$  minimal connected dominating sets that have non-empty intersection with the set  $X$ .*

**Proof.** The set  $X$  cannot have more than two vertices in common with any minimal CDS, since any two elements of  $X$  dominate  $X \cup Y$ . Any minimal CDS that contains exactly one vertex  $x_i$  of  $X$  must contain the vertex  $z$ , to dominate  $y_i$ , and one of the  $t - 1$  neighbors of  $x_i$  (to be connected). There are  $t(t - 1)$  sets of this type. Moreover, each pair of elements of  $X$  dominates  $Y$ . So a minimal CDS can be formed by (any) two elements of  $X$  and any of the elements of  $Y$  (to dominate  $z$ ). There are  $t \frac{t(t-1)}{2}$  such sets. ◀

The hub-vertex  $s$  in  $G_t^k$  must be in any CDS, being a cut-vertex. Therefore, there is no need for the set  $X$  in  $G_t$  to induce a clique, being always dominated by  $s$ . In other words, the counting used in the proof above still holds if each copy of  $G_t$  is replaced by  $G_t - E(X)$  in  $G_t^k$ . Here,  $E(X)$  denotes the set of edges in  $G_t[X]$ . Figure 6b shows  $G_3^3$  without the edges between pairs of element of  $X$  in each copy of  $G_3$ . With the help of Lemma 6.1, we can show:

► **Theorem 6.2.** *The maximum number of minimal CDS in a connected graph of order  $n$  is in  $\Omega(1.4890^n)$ ; an example family of graphs is  $(G_4^k)$ .*

**Proof.** By Lemma 6.1, each copy of the graph  $G_t$  has  $\frac{t^3+t^2}{2} - t$  minimal CDS. There are  $k$  such graphs in  $G_t^k$ , in addition to the vertex  $s$  that connects them all. Every minimal CDS must contain  $s$  and at least one element from  $N(s)$  in each  $G_t$ . Therefore, the total number of minimal CDS in  $G_t^k$  is  $(\frac{t^3+t^2}{2} - t)^k = (\frac{t^3+t^2}{2} - t)^{\frac{n-1}{2t+1}}$ . The claimed lower bound is achieved when  $t = 4$ , which gives a total of  $36^{\frac{n-1}{9}} \in \Omega(1.4890^n)$ . ◀

(a) The graph  $G_4$ .(b) The graph  $G_3^3$ .

■ **Figure 6** How our lower bound examples are composed.

We note that  $G_t^k$  is a  $t$ -degenerate graph that is also bipartite (since the set  $X$  in each copy of  $G_t$  can be an independent set). Furthermore,  $G_3^k$  is planar, as intentionally drawn in Figure 6b. Our general formula (Lemma 6.1) yields that  $G_3^k$  has  $15^{n/7} = \Omega(1.4723^n)$  minimal CDS, improving on the previously mentioned construction for 3-degenerate graphs in [33].

► **Corollary 6.3.** *The maximum number of minimal connected dominating sets in a 3-degenerate bipartite planar graph of order  $n$  is in  $\Omega(1.4723^n)$ .*

Finally,  $G_2^k$  is a 2-degenerate graph of order  $n$  with  $4^{n/5} = \Omega(1.3195^n)$  many minimal CDS, incidentally matching the best known lower bound in cobipartite graphs [9].

## 7 Conclusions and Open Problems

In our paper, we focused on developing an input-sensitive enumeration algorithm for minimal CDS. We achieved some notable progress both on the running time for such enumeration algorithms and with respect to the lower bound examples. However, the gap between lower and upper bound is still quite big, and the natural question to ask here is to bring lower and upper bounds closer; in an optimal setting, both would match. We are working on a further refined version that will bring down the upper bound a bit, but not decisively. This question of non-matching upper and lower bounds is also open for most special graph classes.

One particular such graph class that is studied in this paper is the class of 2-degenerate graphs. We like to suggest to study this graph class also for other enumeration problems, or, more generally, for problems that involve a measure-and-conquer analysis of branching algorithms, because this was the key to break the  $2^n$ -barrier significantly for enumerating minimal CDS with measure-and-conquer, something that seemed to be impossible with other more standard approaches, like putting weights to low-degree vertices.

As we also proved that the extension problem associated to CDS is computationally intractable even on 2-degenerate graphs, it is not that straightforward to analyze our enumeration algorithm with the eyes of output-sensitive analysis. Conversely, should it be possible to find an efficient algorithm for an extension problem, also on special graph classes, then usually polynomial-delay algorithms can be shown; as a recent example in the realm of domination problems, we refer to the enumeration of minimal Roman dominating functions described in [2]. So, in the context of our problem, we can ask: Can we achieve polynomial delay for any enumeration algorithm for minimal CDS? Can we combine this analysis with a good input-sensitive enumeration approach? Notice that the corresponding questions for

enumerating minimal dominating sets are an open question for decades. This is also known as the HITTING SET TRANSVERSAL problem; see [11, 12, 18, 26]. We therefore also presented relations between polynomial-delay enumeration of minimal dominating sets and that of minimal CDS, again explaining the difficulty of the latter question. Finally, we also briefly discussed the possibility of subexponential delay. We propose to discuss this question further also for other enumeration problems when polynomial delay is not achievable, as it might well be a practical solution to know that the delay time is substantially smaller than the time needed to enumerate all solutions, but not polynomial time.

We also discussed parameterized complexity aspects of CONNECTED DOMINATING SET EXTENSION, leaving  $W[3]$ -membership an open question in the general case.

---

## References

- 1 F. N. Abu-Khzam, H. Fernau, B. Gras, M. Liedloff, and K. Mann. Enumerating connected dominating sets. Technical report, Cornell University, ArXiv/CoRR, 2022. doi:10.48550/arXiv.2205.00086.
- 2 F. N. Abu-Khzam, H. Fernau, and K. Mann. Minimal Roman dominating functions: Extensions and enumeration. Technical Report 2204.04765, Cornell University, ArXiv/CoRR, 2022. To appear in the Proceedings of WG 2022. doi:10.48550/arXiv.2204.04765.
- 3 F. N. Abu-Khzam, A. E. Mouawad, and M. Liedloff. An exact algorithm for connected red-blue dominating set. *Journal of Discrete Algorithms*, 9(3):252–262, 2011.
- 4 T. Bläsius, T. Friedrich, J. Lischeid, K. Meeks, and M. Schirneck. Efficiently enumerating hitting sets of hypergraphs arising in data profiling. *Journal of Computer and System Sciences*, 124:192–213, 2022.
- 5 T. Bläsius, T. Friedrich, and M. Schirneck. The complexity of dependency detection and discovery in relational databases. *Theoretical Computer Science*, 900:79–96, 2022.
- 6 H. L. Bodlaender, E. Boros, P. Heggernes, and D. Kratsch. Open problems of the Lorentz workshop “Enumeration algorithms using structure”. Technical Report UU-CS-2015-016, Department of Information and Computing Sciences, Utrecht University, Utrecht, The Netherlands, November 2015.
- 7 K. Casel, H. Fernau, M. Khosravian Ghadikolaei, J. Monnot, and F. Sikora. On the complexity of solution extension of optimization problems. *Theoretical Computer Science*, 904:48–65, 2022.
- 8 J. Chen and F. Zhang. On product covering in 3-tier supply chain models: Natural complete problems for  $W[3]$  and  $W[4]$ . *Theoretical Computer Science*, 363(3):278–288, 2006.
- 9 J.-F. Couturier, P. Heggernes, P. van ’t Hof, and D. Kratsch. Minimal dominating sets in graph classes: Combinatorial bounds and enumeration. In M. Bieliková, G. Friedrich, G. Gottlob, S. Katzenbeisser, and G. Turán, editors, *SOFSEM 2012: Theory and Practice of Computer Science - 38th Conference on Current Trends in Theory and Practice of Computer Science*, volume 7147 of *LNCS*, pages 202–213. Springer, 2012.
- 10 J.-F. Couturier, R. Letourneur, and M. Liedloff. On the number of minimal dominating sets on some graph classes. *Theoretical Computer Science*, 562:634–642, 2015.
- 11 N. Creignou, M. Kröll, R. Pichler, S. Skritek, and H. Vollmer. A complexity theory for hard enumeration problems. *Discrete Applied Mathematics*, 268:191–209, 2019.
- 12 T. Eiter and G. Gottlob. Identifying the minimal transversals of a hypergraph and related problems. *SIAM Journal on Computing*, 24(6):1278–1304, 1995.
- 13 H. Fernau, P. A. Golovach, and M.-F. Sagot. Algorithmic enumeration: Output-sensitive, input-sensitive, parameterized, approximative (Dagstuhl Seminar 18421). *Dagstuhl Reports*, 8(10):63–86, 2018.
- 14 F. V. Fomin, F. Grandoni, and D. Kratsch. Solving Connected Dominating Set faster than  $2^n$ . *Algorithmica*, 52:153–166, 2008.

- 15 F. V. Fomin, F. Grandoni, and D. Kratsch. A measure & conquer approach for the analysis of exact algorithms. *Journal of the ACM*, 56(5), 2009.
- 16 F. V. Fomin, F. Grandoni, A. V. Pyatkin, and A. A. Stepanov. Combinatorial bounds via measure and conquer: Bounding minimal dominating sets and applications. *ACM Transactions on Algorithms*, 5(1):1–17, 2008.
- 17 F. V. Fomin and D. Kratsch. *Exact Exponential Algorithms*. Texts in Theoretical Computer Science. Springer, 2010.
- 18 A. Gainer-Dewar and P. Vera-Licona. The minimal hitting set generation problem: Algorithms and computation. *SIAM Journal of Discrete Mathematics*, 31(1):63–100, 2017.
- 19 P. A. Golovach, P. Heggernes, M. Moustapha Kanté, D. Kratsch, and Y. Villanger. Minimal dominating sets in interval graphs and trees. *Discrete Applied Mathematics*, 216:162–170, 2017.
- 20 P. A. Golovach, P. Heggernes, and D. Kratsch. Enumerating minimal connected dominating sets in graphs of bounded chordality. *Theoretical Computer Science*, 630:63–75, 2016.
- 21 P. A. Golovach, P. Heggernes, D. Kratsch, and R. Saei. Enumeration of minimal connected dominating sets for chordal graphs. *Discrete Applied Mathematics*, 278:3–11, 2020.
- 22 T. W. Haynes, S. T. Hedetniemi, and P. J. Slater. *Fundamentals of Domination in Graphs*, volume 208 of *Monographs and Textbooks in Pure and Applied Mathematics*. Marcel Dekker, 1998.
- 23 R. Impagliazzo, R. Paturi, and F. Zane. Which problems have strongly exponential complexity? *Journal of Computer and System Sciences*, 63(4):512–530, 2001.
- 24 Y. Iwata. A faster algorithm for dominating set analyzed by the potential method. In D. Marx and P. Rossmanith, editors, *Parameterized and Exact Computation - 6th International Symposium, IPEC 2011*, volume 7112 of *LNCS*, pages 41–54. Springer, 2012.
- 25 K. Kangas, P. Kaski, J. H. Korhonen, and M. Koivisto. On the number of connected sets in bounded degree graphs. *Electron. J. Comb.*, 25(4):P4.34, 2018.
- 26 M. M. Kanté, V. Limouzy, A. Mary, and L. Nourine. On the enumeration of minimal dominating sets and related notions. *SIAM Journal of Discrete Mathematics*, 28(4):1916–1929, 2014.
- 27 E. L. Lawler. A note on the complexity of the chromatic number problem. *Information Processing Letters*, 5(3):66–67, 1976.
- 28 D. Lokshtanov, D. Marx, and S. Saurabh. Lower bounds based on the Exponential Time Hypothesis. *EATCS Bulletin*, 105:41–72, 2011.
- 29 D. Lokshtanov, M. Pilipczuk, and S. Saurabh. Below all subsets for minimal connected dominating set. *SIAM Journal of Discrete Mathematics*, 32(3):2332–2345, 2018.
- 30 J. W. Moon and L. Moser. On cliques in graphs. *Israel Journal of Mathematics*, 3:23–28, 1965.
- 31 J. Nederlof, J. M. M. van Rooij, and T. C. van Dijk. Inclusion/exclusion meets measure and conquer. *Algorithmica*, 69(3):685–740, 2014.
- 32 M. Y. Sayadi. *Construction et analyse des algorithmes exacts et exponentiels: énumération input-sensitive. (Design and analysis of exact exponential algorithms: input-sensitive enumeration)*. PhD thesis, University of Lorraine, Nancy, France, 2019.
- 33 M. Y. Sayadi. On the maximum number of minimal connected dominating sets in convex bipartite graphs. Technical Report abs/1908.02174, Cornell University, ArXiv, 2019. [arXiv:1908.02174](https://arxiv.org/abs/1908.02174).
- 34 I. B. Skjærten. Faster enumeration of minimal connected dominating sets in split graphs. Master’s thesis, Department of Informatics, University of Bergen, Norway, June 2017.