Output-Polynomial Enumeration on Graphs of Bounded (Local) Linear MIM-Width*

Petr A. Golovach[†] Pinar Heggernes[†]
Mamadou Moustapha Kanté[‡] Dieter Kratsch[§]
Sigve H. Sæther[†] Yngve Villanger[†]

November 30, 2016

Abstract

The linear induced matching width (LMIM-width) of a graph is a width parameter defined by using the notion of branch-decompositions of a set function on ternary trees. In this paper we study output-polynomial enumeration algorithms on graphs of bounded LMIM-width and graphs of bounded local LMIM-width. In particular, we show that all 1-minimal and all 1-maximal (σ, ρ)-dominating sets, and hence all minimal dominating sets, of graphs of bounded LMIM-width can be enumerated with polynomial (linear) delay using polynomial space. Furthermore, we show that all minimal dominating sets of a unit square graph can be enumerated in incremental polynomial time.

1 Introduction

Enumeration is at the heart of computer science and combinatorics. Enumeration algorithms for graphs and hypergraphs typically deal with listing all vertex subsets or edge subsets satisfying a given property. As the size of the output is often exponential in the size of the input, it is customary to measure the running time of enumeration algorithms in the size of the input plus the size of the output. If the running time of an algorithm is bounded by a polynomial in the size of the input plus the size of the output, then the

*A preliminary version of this paper appeared as an extended abstract in the proceedings of ISAAC 2015. The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP/2007-2013) / ERC Grant Agreement n. 267959. M.M. Kanté and D. Kratsch are supported by French Agency for Research under the GraphEN project (ANR-15-CE-0009)

[†]Department of Informatics, University of Bergen, Norway
[‡]Université Clermont Auvergne, LIMOS, CNRS, Aubiére, France
[§]Université de Lorraine, LITA, Metz, France

algorithm is called output-polynomial. A large number of such algorithms have been given over the last 30 years; many of them solving problems on graphs and hypergraphs [10, 11, 12, 15, 21, 22, 23, 25]. It is also possible to show that certain enumeration problems have no output-polynomial time algorithm unless P = NP [21, 22, 23].

Recently Kanté et al. showed that the famous longstanding open question whether there is an output-polynomial algorithm to enumerate all minimal transversals of a hypergraph, is equivalent to the question whether there is an output-polynomial algorithm to enumerate all minimal dominating sets of a graph [16]. Although the main question remains open, a large number of results have been obtained on graph classes. Output-polynomial algorithms to enumerate all minimal dominating sets exist for graphs of bounded treewidth and of bounded clique-width [9], interval graphs [10], strongly chordal graphs [10], planar graphs [12], degenerate graphs [12], split graphs [16], path graphs [17], permutation graphs [18], line graphs [13, 17, 20], chordal bipartite graphs [14], chordal graphs [19] and graphs of girth at least 7 [13].

In this paper, we extend the above results to graphs of bounded linear maximum induced matching width. Using the notion of branch-decompositions of a set function on ternary trees introduced by Robertson and Seymour, the notion of *maximum induced matching width* (MIM-width) was introduced by Vatshelle [27]. The linear maximum induced matching width (LMIM-width) of a graph is the linearized variant of the MIM-width like path-width is the linearized version of tree-width. (For definitions, see Section 2.) Belmonte and Vatshelle showed that several important graph classes, among them interval, circular-arc and permutation graphs, have bounded LMIM-width [4]. Polynomial-time algorithms solving optimization problems on such graph classes have been studied in [7, 27].

In this paper, we study two ways of using bounded LMIM-width in enumeration algorithms. In Section 3 we study the enumeration problem corresponding to an extended and colored version of the well-known (σ, ρ)-domination problem, asking to enumerate all 1-minimal and all 1-maximal **Red** (σ, ρ)-dominating sets. This includes the enumeration of all minimal (total) dominating sets on graphs of bounded LMIM-width. We establish as our main result an enumeration algorithm with linear delay and polynomial space for this problem. Our algorithm uses the enumeration (and counting) of paths in directed acyclic graphs. In Section 4 we study the enumeration of all minimal dominating sets in unit square graphs. We first show that such graphs have bounded local LMIM-width. A hereditary graph class $\mathcal{G}$ has bounded local LMIM-width if there is a function $f : \mathbb{N} \to \mathbb{N}$ such that the LMIM-width of every graph G in $\mathcal{G}$ is bounded by $f(diam)$ where $diam$ is the diameter of G. The notion of bounded local width has been studied for several width notions and in particular in the area of Bidimensionality [2, 3]. Then we show how to adapt the so-called *flipping method* developed

by Golovach et al. [13] to enumerate all minimal dominating sets of a unit square graph in incremental polynomial time. The flipping method is essentially based on the reverse search technique proposed by Avis and Fukuda in [1] and is tailored for the enumeration of minimal dominating sets in graph classes that satisfy certain properties.

2 Definitions and preliminaries

Graphs. The power set of a set V is denoted by 2^V. For two sets A and B we let $A \backslash B$ be the set $\{x \in A \mid x \notin B\}$, and if X is a subset of a ground set V, we let $\bar{X}$ be the set $V \backslash X$. We often write x to denote the singleton set $\{x\}$. We denote by $\mathbb{N}$ the set of positive or null integers, and let $\mathbb{N}^*$ be $\mathbb{N} \backslash \{0\}$.

A graph G is a pair $(V(G), E(G))$ with $V(G)$ its set of vertices and $E(G)$ its set of edges. An edge between two vertices x and y is denoted by xy (respectively yx). The subgraph of G induced by a subset X of its vertex set is denoted by $G[X]$, and we write $G \backslash X$ to denote the induced subgraph $G[V(G) \backslash X]$. For $F \subseteq E(G)$, we denote by $G - F$ the subgraph $(V(G), E(G) \backslash F)$. The set of vertices that is adjacent to x is denoted by $N_G(x)$, and we let $N_G[x]$ be the set $N_G(x) \cup \{x\}$. For $U \subseteq V(G)$, $N_G[U] = \bigcup_{v \in U} N_G[v]$ and $N_G(U) = N_G[U] \backslash U$. For a vertex x and a positive integer r, $N^r_G[x]$ denotes the set of vertices at distance at most r from x. Clearly, $N^1_G[x] = N_G[x]$. For two disjoint subsets A and B of $V(G)$, let $G[A, B]$ denote the graph with vertex set $A \cup B$ and the edge set $\{uv \in E(G) \mid u \in A, v \in B\}$. Clearly, $G[A, B]$ is a bipartite graph and $\{A, B\}$ is its bipartition. Recall that a set of edges M is an *induced matching* if end-vertices of distinct edges of M are different and not adjacent. We denote by $\mathbf{mim}_G(A, B)$ the size of a maximum induced matching in $G[A, B]$.

Let G be a graph, and let $\mathbf{Red}, \mathbf{Blue} \subseteq V(G)$ such that $\mathbf{Red} \cup \mathbf{Blue} = V(G)$. We refer to the vertices of $\mathbf{Red}$ as the *red* vertices, the vertices of $\mathbf{Blue}$ as the *blue* vertices, and we say that G together with given sets $\mathbf{Red}$ and $\mathbf{Blue}$ is a *colored graph*. For simplicity, whenever we say that G is a colored graph, it is assumed that the sets $\mathbf{Red}$ and $\mathbf{Blue}$ are given. Notice that $\mathbf{Red}$ and $\mathbf{Blue}$ are not necessarily disjoint. In particular, it can happen that $\mathbf{Red} = \mathbf{Blue} = V(G)$; a non-colored graph G can be seen as a colored graph with $\mathbf{Red} = \mathbf{Blue} = V(G)$. We deal with colored graphs because our algorithm for unit-square graphs requires as a subroutine an algorithm that takes as input a colored graph and enumerates all minimal subsets of red vertices that dominate the blue vertices.

A graph G is an *(axis-parallel) unit square* graph if it is an intersection graph of unit squares in the plane with their sides parallel to the coordinate axis. These graphs are also known as the graphs of cubicity 2. We use the following equivalent definition, see e.g. [8], in which each vertex v of G is

represented by a point in $\mathbb{R}^2$. A graph G is a unit square graph if there is a function $f\colon V(G) \to \mathbb{R}^2$ such that two vertices $u, v \in V(G)$ are adjacent in G if and only if $\|f(u) - f(v)\|_\infty < 1$, where the norm $\|\|_\infty$ is the L_∞ norm. For a vertex $v \in V(G)$, we let $x_f(v)$ and $y_f(v)$ denote the x and y-coordinate of $f(v)$ respectively. We say that the point $(x_f(v), y_f(v))$ *represents* v. The function f is called a *realization* of the unit square graph. It is straightforward to see that for any unit square graph G, there is a realization $f\colon V(G) \to \mathbb{Q}^2$. We always assume that a unit square graph is given with its realization. It is NP-hard to recognize unit square graphs [6]. We refer to the survey of Brandstädt, Le and Spinrad [5] for the definitions of all other graph classes mentioned in our paper.

Enumeration. Let $\mathcal{D}$ be a family of subsets of the vertex set of a given graph G on n vertices and m edges. An *enumeration algorithm* for $\mathcal{D}$ lists the elements of $\mathcal{D}$ without repetitions. The running time of an enumeration algorithm $\mathcal{A}$ is said to be *output-polynomial* if there is a polynomial $p(x, y)$ such that all the elements of $\mathcal{D}$ are listed in time bounded by $p((n+m), |\mathcal{D}|)$. Assume now that $D_1, \ldots, D_\ell$ are the elements of $\mathcal{D}$ enumerated in the order in which they are generated by $\mathcal{A}$. Let us denote by $T(\mathcal{A}, i)$ the time $\mathcal{A}$ requires until it outputs D_i, also $T(\mathcal{A}, \ell + 1)$ is the time required by $\mathcal{A}$ until it stops. Let $delay(\mathcal{A}, 1) = T(\mathcal{A}, 1)$ and $delay(\mathcal{A}, i) = T(\mathcal{A}, i) - T(\mathcal{A}, i-1)$. The *delay* of $\mathcal{A}$ is $\max\{delay(\mathcal{A}, i)\}$. Algorithm $\mathcal{A}$ runs in *incremental polynomial* time if there is a polynomial $p(x, i)$ such that $delay(\mathcal{A}, i) \leqslant p(n + m, i)$. Furthermore $\mathcal{A}$ is a *polynomial delay* algorithm if there is a polynomial $p(x)$ such that the delay of $\mathcal{A}$ is at most $p(n + m)$. Finally $\mathcal{A}$ is a *linear delay* algorithm if $delay(\mathcal{A}, 1)$ is bounded by a polynomial in $n+m$ and $delay(\mathcal{A}, i)$ is bounded by a linear function in $|D_{i-1}| + |D_i|$.

Linear maximum induced matching width. The notion of the *maximum induced matching width* was introduced by Vatshelle [27] (see also [4]). We will give the definition in terms of colored graphs and restrict ourselves to the case of linear maximum induced matching width. Let G be a colored n-vertex graph with $n \geqslant 2$ and let $x_1, \ldots, x_n$ be a linear ordering of its vertex set. For each $1 \leqslant i \leqslant n$, we let $A_i = \{x_1, x_2, \ldots x_i\}$ and $\bar{A}_i = \{x_{i+1}, x_{i+2}, \ldots x_n\}$. The *maximum induced matching width* (*MIM-width* for short) of $x_1, \ldots, x_n$ is

$$\max_{1 \leqslant i \leqslant n-1} \max\{\mathbf{mim}_G(A_i \cap \mathbf{Red}, \bar{A}_i \cap \mathbf{Blue}), \mathbf{mim}_G(A_i \cap \mathbf{Blue}, \bar{A}_i \cap \mathbf{Red})\}.$$

Notice that if $\mathbf{Red} = \mathbf{Blue} = V(G)$, *i.e.*, if G is an uncolored graph, the MIM-width of $x_1, \ldots, x_n$ is

$$\max\{\mathbf{mim}_G(A_i, \overline{A}_i) | 1 \leqslant i \leqslant n-1\}.$$

Consequently, The *linear maximum induced matching width* (*LMIM-width*) of G, denoted by $\mathbf{lmimw}(G)$, is the minimum value of the MIM-width taken over all linear orderings of G.

Belmonte and Vatshelle [4] proved that several important graph classes have bounded linear maximum induced matching width.

Theorem 1. *For each of the following graph classes: interval graphs, permutation graphs, circular-arc graphs, circular permutation graphs, trapezoid graphs, convex graphs, and for fixed k, k-polygon graphs, Dilworth-k graphs and complements of k-degenerate graphs, there is a constant c such that $\mathbf{lmimw}(G) \leqslant c$ for any graph G from the class. Moreover, the corresponding linear ordering of the vertices of MIM-width at most c can be found in polynomial time.*

For example, the LMIM-width of an interval graph is 1 and the LMIM-width of a circular-arc and of a permutation graph is at most 2. Before continuing, let us show that if a graph class $\mathcal{G}$ has LMIM-width c, then the graph class $\mathcal{G}'$ obtained from the graphs in $\mathcal{G}$ by partitioning their vertex set into **Blue** and **Red** also has LMIM-width c.

Proposition 2. *If a colored graph G' is obtained from a non-colored graph G, then $\mathbf{lmimw}(G') \leqslant \mathbf{lmimw}(G)$.*

Proof. Let $A \subset V(G)$. For the colored graph G',

$$\mathbf{mim}_{G'}(A, \overline{A}) \leqslant \max\{\mathbf{mim}_{G'}(A \cap \mathbf{Red}, \overline{A} \cap \mathbf{Blue}), \mathbf{mim}_{G'}(A \cap \mathbf{Blue}, \overline{A} \cap \mathbf{Red})\}.$$

Because

$$\mathbf{mim}_{G'}(A \cap \mathbf{Red}, \overline{A} \cap \mathbf{Blue}) = \mathbf{mim}_{G}(A \cap \mathbf{Red}, \overline{A} \cap \mathbf{Blue}) \leqslant \mathbf{mim}_{G}(A, \overline{A})$$

and

$$\mathbf{mim}_{G'}(A \cap \mathbf{Blue}, \overline{A} \cap \mathbf{Red}) = \mathbf{mim}_{G}(A \cap \mathbf{Blue}, \overline{A} \cap \mathbf{Red}) \leqslant \mathbf{mim}_{G}(A, \overline{A}),$$

$\mathbf{mim}_{G'}(A, \overline{A}) \leqslant \mathbf{mim}_{G}(A, \overline{A})$ and the claim follows. □

We say that a graph class $\mathcal{G}$ has *locally bounded LMIM-width* if there is a function $f : \mathbb{N} \to \mathbb{N}$ such that for any $G \in \mathcal{G}$ and every $u \in V(G)$, $\mathbf{lmimw}(G[N_G^r[u]]) \leqslant f(r)$ for all $r \in \mathbb{N}$.

(σ, ρ)-domination. The *(σ, ρ)-dominating set* notion was introduced by Telle and Proskurowski [26] as a generalization of dominating sets. Indeed, many NP-hard domination type problems such as the problems d-Dominating Set, Independent Dominating Set and Total Dominating Set are special cases of the (σ, ρ)-Dominating Set Problem. See [7, Table 1] for more examples. For technical reasons, we introduce **Red** (σ, ρ)-domination. Let σ and ρ be subsets of $\mathbb{N}$. Throughout this paper it is assumed that σ

and ρ are finite or co-finite. Notice that it can happen that σ is finite and ρ is co-finite and vice versa. We say that a subset D of $V(G)$ (σ,ρ)-*dominates* a vertex u if

$$|N_G(u) \cap D| \in \begin{cases} \sigma & \text{if } u \in D, \\ \rho & \text{if } u \notin D. \end{cases}$$

It is said that D (σ,ρ)-*dominates* $U \subseteq V(G)$ if D (σ,ρ)-dominates every vertex of U.

Let G be a colored graph. A set of vertices $D \subseteq \mathbf{Red}$ is a **Red** (σ,ρ)-*dominating set* if D (σ,ρ)-dominates **Blue**. If $\mathbf{Red} = \mathbf{Blue} = V(G)$, then a **Red** (σ,ρ)-dominating set is a (σ,ρ)-*dominating set*.

Notice that if $\sigma = \mathbb{N}$ and $\rho = \mathbb{N}^*$, then a set $D \subseteq V(G)$ (σ,ρ)-dominates a vertex u if $u \in D$ or u is adjacent to a vertex of D, *i.e.*, the notion of (σ,ρ)-domination coincides with the classical domination in this case. Whenever we consider this case, we simply write that a set D dominates a vertex or set and D is a (**Red**) dominating set omitting (σ,ρ).

A **Red** (σ,ρ)-dominating set D of a graph G is said *minimal* if for any proper subset $D' \subset D$, D' is not a **Red** (σ,ρ)-dominating set, and we say that D is 1-*minimal* if for each vertex x in D, $D\backslash x$ is not a **Red** (σ,ρ)-dominating set. Respectively, a **Red** (σ,ρ)-dominating set D is *maximal* if for any D' such that $D \subset D' \subseteq \mathbf{Red}$, D' is not a **Red** (σ,ρ)-dominating set and D is 1-*maximal* if for each vertex x in $\mathbf{Red}\backslash D$, $D \cup \{x\}$ is not a **Red** (σ,ρ)-dominating set. Clearly, every minimal or maximal **Red** (σ,ρ)-dominating set is 1-minimal or 1-maximal, respectively, but not the other way around because the converse is not true for arbitrary σ and ρ. Observe however that every (**Red**) 1-minimal (total) dominating set is also a (**Red**) minimal (total) dominating set. In Section 3 we enumerate only 1-minimal and 1-maximal **Red** (σ,ρ)-dominating sets.

Because our aim is to enumerate 1-minimal or 1-maximal **Red** (σ,ρ)-dominating sets, we need some certificate that a considered set is 1-minimal or 1-maximal respectively. Let D be a **Red** (σ,ρ)-dominating set of a colored graph G. For a vertex $u \in D$, we say that the vertex $v \in \mathbf{Blue}$ is its *certifying vertex* (or a *certificate*) if v is not (σ,ρ)-dominated by $D\backslash\{u\}$, *i.e.*,

$$|N_G(v) \cap (D\backslash\{u\})| \notin \begin{cases} \sigma & \text{if } v \in (D\backslash\{u\}) \cap \mathbf{Blue}, \\ \rho & \text{if } v \in \mathbf{Blue}\backslash(D\backslash\{u\}). \end{cases}$$

Respectively, for a vertex $u \in \mathbf{Red}\backslash D$, the vertex $v \in \mathbf{Blue}$ is its *certifying vertex* (or a *certificate*) if v is not (σ,ρ)-dominated by $D \cup \{u\}$.

Notice that because D is a **Red** (σ,ρ)-dominating set, if v is a certificate for u, then $v \in N_G[u]$. Observe also that a vertex can be a certificate for

many vertices and it can be a certificate for itself. Notice that in the case of the classical domination, certificates are usually called *privates* because a vertex is always a certificate for exactly one vertex, including itself. It is straightforward to show the following.

Lemma 3. *A set $D \subseteq$ **Red** is a 1-minimal* **Red** (σ, ρ)*-dominating set of a colored graph G if and only if each vertex $u \in D$ has a certificate. Furthermore, D is a 1-maximal* **Red** (σ, ρ)*-dominating set of G if and only if each vertex $u \in$ **Red**$\backslash D$ has a certificate.*

3 Enumerations for graphs of bounded LMIM-width

In this section we prove the following.

Theorem 4. *Let (σ, ρ) be a pair of finite or co-finite subsets of $\mathbb{N}$ and let c be a positive integer. For a colored graph G given with a linear ordering of $V(G)$ of LMIM-width at most c, one can, after a pre-processing in time $n^{O(c)}$, count in time bounded by $n^{O(c)}$, and enumerate with linear delay, all 1-minimal (or 1-maximal)* **Red** (σ, ρ)*-dominating sets of G.*

As a corollary of Theorems 1 and 4, and Proposition 2 we have the following.

Corollary 5. *Let (σ, ρ) be a pair of finite or co-finite subsets of $\mathbb{N}$. Then, for every colored graph G in one of the following graph classes, we can count in polynomial time, and enumerate with linear delay all 1-minimal (or 1-maximal)* **Red** (σ, ρ)*-dominating sets of G: interval graphs, permutation graphs, circular-arc graphs, circular permutation graphs, trapezoid graphs, convex graphs, and for fixed k, k-polygon graphs, Dilworth-k graphs and complements of k-degenerate graphs.*

The following corollary improves some known results in the enumeration of minimal transversals of some geometric hypergraphs (see e.g. [24]).

Corollary 6. *For every hypergraph $\mathcal{H}$ being an interval hypergraph or a circular-arc hypergraph one can count in polynomial time, and enumerate with linear delay, all minimal transversals of $\mathcal{H}$.*

Proof. For any of the considered hypergraphs, its incidence graph is a subgraph of an interval or a circular-arc graph. If we color the vertices of the hypergraph in **Red** and the hyperedges in **Blue**, then X is a minimal transversal in the hypergraph if and only if it is a **Red** dominating set. □

The remaining part of the section is devoted to the proof of Theorem 4. In Section 3.1 we give some technical definitions and lemmas that are important for the definition of the DAG whose maximal paths correspond

to the desired sets. In Section 3.2 we define the DAG whose maximal paths correspond to the 1-minimal **Red** (σ, ρ)-dominating sets and then show that it can be constructed in polynomial time. We also recall how to count in polynomial time, and enumerate with linear delay the maximal paths of a DAG. We then explain in Section 3.3 how the construction of Section 3.2 can be rewritten for 1-maximal **Red** (σ, ρ)-dominating sets.

3.1 Technical Definitions

First because σ and ρ can be infinite, we need a finite way to check if a vertex is (σ, ρ)-dominated. Let $d(\mathbb{N}) = 0$. For every finite set $\mu \subseteq \mathbb{N}$, let $d(\mu) := 1 + \max\{a \mid a \in \mu\}$, and for every co-finite set $\mu \subseteq \mathbb{N}$, let $d(\mu) := 1 + \max\{a \mid a \in \mathbb{N}\backslash\mu\}$. For finite or co-finite subsets σ and ρ of $\mathbb{N}$, we let $d(\sigma, \rho) := \max(d(\sigma), d(\rho))$. Given a subset D of **Red**, we can check if D is a **Red** (σ, ρ)-dominating set by computing $|D \cap N_G(x)|$ up to $d(\sigma, \rho)$ for each vertex x in **Blue** [7]. We need the following properties of certificates.

Lemma 7. *Let D be a* **Red** *(σ, ρ)-dominating set of a colored graph G. If v is a certificate for $u \in D$, then $v = u$ or v is a certificate for all vertices of $N_G(v) \cap D$. If v is a certificate for $u \in \overline{D} \cap$* **Red***, then $v = u$ or v is a certificate for all vertices of $N_G(v) \cap \overline{D} \cap$* **Red***.*

Proof. Let $v \neq u$ be a certificate for $u \in D$ and let $D' := D\backslash\{u\}$. If $v \in D'$ and $|N_G(v) \cap D'| \notin \sigma$, then for any $w \in N_G(v) \cap D$, $|N_G(v) \cap (D\backslash\{w\})| = |N_G(v) \cap D'| \notin \sigma$. If $v \notin D'$ and $|N_G(v) \cap D'| \notin \rho$, then for any $w \in N_G(v) \cap D$, $|N_G(v) \cap (D\backslash\{w\})| = |N_G(v) \cap D'| \notin \rho$. The second claim can be proved by similar arguments. □

We define $\sigma^* := \sigma\backslash\rho$ and $\rho^* := \rho\backslash\sigma$. Let also $\sigma^- := \{i \in \sigma \mid i - 1 \notin \sigma\}$, $\rho^- := \{i \in \rho \mid i - 1 \notin \rho\}$, $\sigma^+ := \{i \in \sigma \mid i + 1 \notin \sigma\}$ and $\rho^+ := \{i \in \rho \mid i + 1 \notin \rho\}$. By the definitions, we have the following property.

Lemma 8. *The sets $\sigma^*, \rho^*, \sigma^-, \rho^-, \sigma^+, \rho^+$ are finite or co-finite. Also, $d(\sigma^*, \rho^*) \leqslant d(\sigma, \rho)$, $d(\sigma^-, \rho^-) \leqslant d(\sigma, \rho) + 1$ and $d(\sigma^+, \rho^+) \leqslant d(\sigma, \rho) + 1$.*

By the definition of certificates, we have the next easy lemma.

Lemma 9. *Let D be a* **Red** *(σ, ρ)-dominating set of a colored graph G and let $u \in$* **Red** *and $v \in$* **Blue** *be distinct vertices of G. If $u \in D$, then v is a certificate for u if and only if*

$$|N_G(v) \cap D| \in \begin{cases} \sigma^- & \text{if } v \in D \\ \rho^- & \text{if } v \notin D. \end{cases}$$

If $u \notin D$, then v is a certificate for u if and only if

$$|N_G(v) \cap D| \in \begin{cases} \sigma^+ & \text{if } v \in D \\ \rho^+ & \text{if } v \notin D. \end{cases}$$

A blue vertex $v \in D$ is a certificate for itself if and only if $|N_G(v) \cap D| \in \sigma^*$. A red vertex $v \notin D$ is a certificate for itself if and only if it is blue and $|N_G(v) \cap D| \in \rho^*$.

Let $d \in \mathbb{N}$ and let A be a subset of the vertex set of a colored graph G. Two red subsets X and Y of A are *d-neighbor equivalent* w.r.t. A, denoted by $X \equiv_A^d Y$, if for all $x \in \bar{A} \cap \mathbf{Blue}$ we have

$$\min(d, |X \cap N_G(x)|) = \min(d, |Y \cap N_G(x)|).$$

It is not hard to check that $\equiv_A^d$ is an equivalence relation and let us denote by $nec(\equiv_A^d)$ the number of equivalence classes of $\equiv_A^d$. Belmonte and Vatshelle [4] proved the following lemma restated in our setting.

Lemma 10 ([4])**.** *Let $d \in \mathbb{N}$ and let A be a subset of the vertex set of a colored graph G such that $\mathbf{mim}_G(A \cap \mathbf{Red}, \bar{A} \cap \mathbf{Blue}) \leqslant k$. Then $nec(\equiv_A^d) \leqslant n^{d \cdot k}$.*

The next lemma is used to bound the number of information we have to store at each node of the DAG, which will consequently imply, combined with Lemma 10, that the size of the DAG is polynomial in the size of G.

Lemma 11 ([4])**.** *Let G be a colored graph and let A be a subset of $V(G)$. Then, $\mathbf{mim}_G(A \cap \mathbf{Red}, \bar{A} \cap \mathbf{Blue}) \leqslant k$ if and only if for every blue subset S of $\bar{A}$ there is $C \subseteq S$ such that $N(C) \cap (A \cap \mathbf{Red}) = N(S) \cap (A \cap \mathbf{Red})$ and $|C| \leqslant k$.*

3.2 Constructing the DAG for 1-minimal sets

Throughout this section we let (σ, ρ) be a fixed pair of finite or co-finite subsets of $\mathbb{N}$ and we let G be a fixed n-vertex colored graph with $n \geqslant 2$. Let also $x_1, \ldots, x_n$ be a fixed linear ordering of the vertex set of G such that the MIM-width of $x_1, \ldots, x_n$ is bounded by a constant c. Furthermore, for all $i \in \{1, 2, \ldots, n\}$, we let $A_i = \{x_1, x_2, \ldots x_i\}$ and $\bar{A}_i = \{x_{i+1}, x_{i+2}, \ldots x_n\}$. We furthermore let $d = d(\sigma, \rho)$.

We will follow the same idea as in [7] where a minimum (or a maximum) (σ, ρ)-dominating set is computed, and we need for that to recall some definitions and lemmas (restated in our setting) proved in [7]. For every $i \in \{1, \ldots, n\}$ and every subset X of $A_i \cap \mathbf{Red}$, we denote by $rep_{A_i}^d(X)$ the lexicographically smallest set $R \subseteq A_i \cap \mathbf{Red}$ such that $|R|$ is minimised and $R \equiv_{A_i}^d X$. Notice that it can happen that $R = \varnothing$.

Lemma 12 ([7])**.** *For every $i \in \{1, \ldots, n\}$, one can compute a list LR_i^d containing all representatives* w.r.t. $\equiv_{A_i}^d$ *in time $O(nec(\equiv_{A_i}^d) \cdot \log(nec(\equiv_{A_i}^d)) \cdot n^2)$. One can also compute a data structure that given a set $X \subseteq A_i \cap \mathbf{Red}$ in time $O(\log(nec(\equiv_{A_i}^d)) \cdot |X| \cdot n)$ allows us to find a pointer to $rep_{A_i}^d(X)$ in LR_i^d. Similar statements hold for the list $LR_{\bar{i}}^d$ containing all representatives* w.r.t. $\equiv_{\bar{A}_i}^d$.

We will define a DAG, denoted by $DAG(G)$, the maximal paths of which correspond exactly to the 1-minimal **Red** (σ,ρ)-dominating sets of G. Let us first explain the nodes and arcs of the DAG. For a 1-minimal **Red** (σ,ρ)-dominating set D and $1 \leqslant j \leqslant n$, let $D_j = D \cap \{x_1,\ldots,x_j\}$. The arcs of $DAG(G)$ describe how we construct D_{j+1} from D_j. Since each vertex of D must have a certificate, we should ensure when $D_{j+1} = D_j \cup \{x_{j+1}\}$ that x_{j+1} has a certificate and also no vertex in D_j loses its certificate. Now, x_{j+1} either has a certificate in $A_{j+1} \cap$ **Blue** or in $\bar{A}_{j+1} \cap$ **Blue**. However, by Lemma 11 we know that if x_{j+1} has a certificate in $A_{j+1} \cap$ **Blue** (or in $\bar{A}_{j+1} \cap$ **Blue**), then there is a set C (or C') of size at most c so that it has a certificate in C (or in C'). In the nodes of $DAG(G)$ we will guess these sets. In order to polynomially bound the number of nodes, we need also to classify the different sets D_j into a polynomial number of classes, and also identify which such sets can be extended to 1-minimal **Red** (σ,ρ)-dominating sets. But because only adjacency is important for domination problems, we will use Lemma 12 to guess such classes. Therefore, a node of $DAG(G)$ will be a tuple (R,R',C,C',j) where $1 \leqslant j \leqslant n$ identifies the vertices to add in the partial solution, (R,R') are the equivalence classes of D_j and of $D\backslash D_j$ *w.r.t.* $\equiv^d_{A_j}$ and $\equiv^d_{\bar{A}_j}$ respectively, and C (resp. C') the guessing of the certificates for vertices in $D\backslash D_j$ (resp. D_j). The arcs describe the extensions of such partial solutions and an arc from (R_1,R'_1,C_1,C'_1,j) to $(R_p,R'_p,C_p,C'_p,j+p)$ means that $D_{j+p} = D_j \cup \{x_{j+p}\}$, all vertices x_{j+i}, for $1 \leqslant i \leqslant p$ are (σ,ρ)-dominated by any extension of D_{j+p}, and x_{j+p} has a certificate and will continue to have a certificate in any extension of D_{j+p} into a 1-minimal **Red** (σ,ρ)-dominating set.

For $1 \leqslant j \leqslant n$ and $C \subseteq A_j \cap$ **Blue** (or $C \subseteq \bar{A}_j \cap$ **Blue**) we denote by $\mathcal{SG}_j(C)$ (or by $\mathcal{GG}_j(C)$) the set X obtained from C if we initially set $X = C$ and recursively apply the following rule: let x be the greatest (or smallest) vertex in X such that $N(X\backslash\{x\}) \cap (\bar{A}_j \cap$ **Red**$) = N(X) \cap (\bar{A}_j \cap$ **Red**$)$ (or $N(X\backslash\{x\}) \cap (A_j \cap$ **Red**$) = N(X) \cap (A_j \cap$ **Red**$)$) and set $X = X\backslash\{x\}$. Notice that $\mathcal{SG}_j(C)$ and $\mathcal{GG}_j(C)$ are both uniquely determined, and both have sizes bounded by c from Lemma 11. Observe also that if $C \subseteq A_j \cap$ **Blue** (or $C \subseteq \bar{A}_j \cap$ **Blue**), then $\mathcal{SG}_\ell(C \cup \{x_\ell\}) = \mathcal{SG}_\ell(\mathcal{SG}_j(C) \cup \{x_\ell\})$ for all $\ell > j$ (or $\mathcal{GG}_\ell(C \cup \{x_\ell\}) = \mathcal{GG}_\ell(\mathcal{GG}_j(C) \cup \{x_\ell\})$ for all $\ell \leqslant j$). The constructors $\mathcal{SG}_j$ and $\mathcal{GG}_j$ are used to canonically choose certificates in order to avoid redundancies (see the conditions (1.2) and (2.2) below).

Let $1 \leqslant j < n$ and let $(R_j,R'_j,C_j,C'_j) \in LR^d_j \times LR^d_{\bar{j}} \times 2^{A_j \cap \textbf{Blue}} \times 2^{\bar{A}_j \cap \textbf{Blue}}$ and $(R_{j+1},R'_{j+1},C_{j+1},C'_{j+1}) \in LR^d_{j+1} \times LR^d_{\overline{j+1}} \times 2^{A_{j+1} \cap \textbf{Blue}} \times 2^{\bar{A}_{j+1} \cap \textbf{Blue}}$. There is an *$\varepsilon$-arc-1* from (R_j,R'_j,C_j,C'_j) to $(R_{j+1},R'_{j+1},C_{j+1},C'_{j+1})$ if

(1.1) $R_j \equiv^d_{A_{j+1}} R_{j+1}$ and $R'_j \equiv^d_{\bar{A}_j} R'_{j+1}$, and

(1.2) if ($x_{j+1} \notin$ **Blue** or ($x_{j+1} \in$ **Blue** and $|N(x_{j+1}) \cap (R_j \cup R'_{j+1})| \in \rho$ and

$|N(x_{j+1}) \cap (R_j \cup R'_{j+1})| \notin \rho^-)$) then ($C_{j+1} = \mathcal{SG}_{j+1}(C_j)$ and $C'_j = \mathcal{GG}_j(C'_{j+1})$), otherwise we should have ($|N(x_{j+1}) \cap (R_j \cup R'_{j+1})| \in \rho^-$) and

(1.2.a) if $N(x_{j+1}) \cap (\bar{A}_{j+1} \cap \mathbf{Red}) \neq \varnothing$, then $C_{j+1} = \mathcal{SG}_{j+1}(C_j \cup \{x_{j+1}\})$, else $C_{j+1} = \mathcal{SG}_{j+1}(C_j)$, and

(1.2.b) if $N(x_{j+1}) \cap (A_j \cap \mathbf{Red}) \neq \varnothing$, then $C'_j = \mathcal{GG}_j(C'_{j+1} \cup \{x_{j+1}\})$, else $C'_j = \mathcal{GG}_j(C'_{j+1})$.

There is an *ε-arc-2* from (R_j, R'_j, C_j, C'_j) to $(R_{j+1}, R'_{j+1}, C_{j+1}, C'_{j+1})$ if

(2.1) $R_{j+1} \equiv^d_{A_{j+1}} (R_j \cup \{x_{j+1}\})$, $R'_j \equiv^d_{\bar{A}_j} (R'_{j+1} \cup \{x_{j+1}\})$, $x_{j+1} \in \mathbf{Red}$, ($|N(x_{j+1}) \cap (R_j \cup R'_{j+1})| \in \sigma$ if $x_{j+1} \in \mathbf{Blue}$), and

(2.2) if ($x_{j+1} \notin \mathbf{Blue}$ or ($x_{j+1} \in \mathbf{Blue}$ and $|N(x_{j+1}) \cap (R_j \cup R'_{j+1})| \notin \sigma^-$)), then ($C_{j+1} = \mathcal{SG}_{j+1}(C_j)$ and $C'_j = \mathcal{GG}_j(C'_{j+1})$), otherwise we should have ($|N(x_{j+1}) \cap (R_j \cup R'_{j+1})| \in \sigma^-$) and

(2.2.a) if $N(x_{j+1}) \cap (\bar{A}_{j+1} \cap \mathbf{Red}) \neq \varnothing$, then $C_{j+1} = \mathcal{SG}_{j+1}(C_j \cup \{x_{j+1}\})$, else $C_{j+1} = \mathcal{SG}_{j+1}(C_j)$, and

(2.2.b) if $N(x_{j+1}) \cap (A_j \cap \mathbf{Red}) \neq \varnothing$, then $C'_j = \mathcal{GG}_j(C'_{j+1} \cup \{x_{j+1}\})$, else $C'_j = \mathcal{GG}_j(C'_{j+1})$, and

(2.3) either ($N(x_{j+1}) \cap (C_j \cup C'_{j+1}) \neq \varnothing$) or (($x_{j+1} \in \mathbf{Blue}$ and $|N(x_{j+1}) \cap (R_j \cup R'_{j+1})| \in \sigma^*$).

The nodes of $DAG(G)$.

$$(R, R', C, C', i) \in LR^d_i \times LR^d_{\bar{i}} \times 2^{A_i \cap \mathbf{Blue}} \times 2^{\bar{A}_i \cap \mathbf{Blue}} \times \{1, \ldots, n\}$$

is a node of $DAG(G)$ whenever $x_i \in \mathbf{Red}$, $C = \mathcal{SG}_i(C)$ and $C' = \mathcal{GG}_i(C')$. We call i the *index* of (R, R', C, C', i). Finally $s = (\varnothing, \varnothing, \varnothing, \varnothing, 0)$ is the *source node* and $t = (\varnothing, \varnothing, \varnothing, \varnothing, n+1)$ is the *terminal node* of $DAG(G)$.

The arcs of $DAG(G)$. There is an arc from the node $(R_0, R'_0, C_0, C'_0, j)$ to the node $(R_p, R'_p, C_p, C'_p, j+p)$ with $1 \leqslant j < j+p \leqslant n$ if there exist tuples (R_1, R'_1, C_1, C'_1), …, $(R_{p-1}, R'_{p-1}, C_{p-1}, C'_{p-1})$ such that (1) for each $1 \leqslant i \leqslant p-1$,

$$(R_i, R'_i, C_i, C'_i) \in LR^d_{j+i} \times LR^d_{\overline{j+i}} \times 2^{A_{j+i} \cap \mathbf{Blue}} \times 2^{\bar{A}_{j+i} \cap \mathbf{Blue}}$$

and there is an ε-arc-1 from $(R_{i-1}, R'_{i-1}, C_{i-1}, C'_{i-1})$ to (R_i, R'_i, C_i, C'_i), and (2) there is an ε-arc-2 from $(R_{p-1}, R'_{p-1}, C_{p-1}, C'_{p-1})$ to (R_p, R'_p, C_p, C'_p).

Let $u = (R, R', C, C', j)$ be a node and let $S = \{x \in (A_j \cap \mathbf{Blue}) \backslash \{x_j\} \mid N(x) \cap (\bar{A}_j \cap \mathbf{Red}) \neq \varnothing$ and $|N(x) \cap (\{x_j\} \cup R')| \in \rho^-\}$. There is an arc from the source node to u if the following three conditions are satisfied

(S1) $\{x_j\} \equiv^d_{A_j} R$ and $(\{x_j\} \cup R')$ (σ, ρ)-dominates $A_j \cap \textbf{Blue}$,

(S2) $C = \mathcal{SG}_j(S \cup \{x_j\})$ whenever ($x_j \in \textbf{Blue}$ and $|N(x_j) \cap R'| \in \sigma^-$), otherwise $C = \mathcal{SG}_j(S)$, and

(S3) either $(N(x_j) \cap (C' \cup C) \neq \emptyset)$ or ($x_j \in \textbf{Blue}$ and $|N(x_j) \cap R'| \in \sigma^*$).

There is an arc from a node (R, R', C, C', j) to the terminal node if

(T1) $|N(x) \cap R| \in \rho$ for each $x \in \bar{A}_{j+1} \cap \textbf{Blue}$, and

(T2) $C' = \mathcal{GG}_j(\{x \in \bar{A}_j \cap \textbf{Blue} \mid N(x) \cap (A_j \cap \textbf{Red}) \neq \emptyset$ and $|N(x) \cap R| \in \rho^-\})$.

Lemma 13. *$DAG(G)$ is a DAG and can be constructed in time $n^{O(c \cdot d)}$.*

Proof. An arc is always oriented from a node (R, R', C, C', j) to $(\hat{R}, \hat{R}', \hat{C}, \hat{C}', j+p)$ with $p \geqslant 1$. Therefore, we cannot create circuits, *i.e.*, $DAG(G)$ is a DAG.

For each index $1 \leqslant i \leqslant n$ and each node (R, R', C, C', i) of index i we know by Lemma 11 that $|C|, |C'| \leqslant c$. Hence, the number of nodes of $DAG(G)$ of index i is bounded by

$$\begin{aligned} |LR^d_i| \cdot |LR^d_{\bar{i}}| \cdot n^c \cdot n^c &\leqslant n^{c \cdot d} \cdot n^{c \cdot d} \cdot n^{2c} \\ &= n^{2c(d+1)} \end{aligned}$$

since $|LR^d_i| = nec(\equiv^d_{A_i}) \leqslant n^{d \cdot c}$ by Lemma 10, and c and d are both constants. From Lemma 12 one can access to LR^d_i and $LR^d_{\bar{i}}$ for each $1 \leqslant i \leqslant n$ in time $n^{O(c \cdot d)}$. Then, one can compute the set of nodes in time $n^{O(c \cdot d)}$ since one can check whether $C = \mathcal{SG}_j(C)$ (or $C' = \mathcal{GG}_j(C')$) in time $O(n^3)$.

For computing the arcs, we first notice that each of the conditions (S1)-(S3), (T1)-(T2), and of the conditions of ε-arc-1 and of ε-arc-2 can be checked in at most $O(n^3)$ time because each is based on computing the neighbors of a set and/or the use of Lemma 12 to compute representatives.

Now, constructing the arcs from the source node can be done in $n^{O(c \cdot d)}$ time since it suffices to check for each node (R, R', C, C', j) if conditions (S1)-(S3) are satisfied, which can be done in $O(n^3)$ time. Similarly, since the conditions (T1) and (T2) can be checked in $O(n^3)$ time, the incoming arcs to the terminal node can be constructed in $n^{O(c \cdot d)}$ time.

Now, to construct an arc from $(R_0, R'_0, C_0, C'_0, j)$ to $(R_p, R'_p, C_p, C'_p, j+p)$ we do as follows. For $0 \leqslant i \leqslant p-1$ we let $\mathcal{F}_i$ be a queue and we put (R_0, R'_0, C_0, C'_0) in $\mathcal{F}_0$. Now, for $1 \leqslant i \leqslant p-1$, pull $(R_{i-1}, R'_{i-1}, C_{i-1}, C'_{i-1})$ from $\mathcal{F}_{i-1}$, and for each (R, R', C, C') with $R \in LR^d_{j+i}$, $R' \in LR^d_{\overline{j+i}}$, $C \subseteq A_{j+i} \cap \textbf{Blue}$, $C' \subseteq \bar{A}_{j+i} \cap \textbf{Blue}$ with $|C|, |C'| \leqslant c$ such that there is an ε-arc-1 from $(R_{i-1}, R'_{i-1}, C_{i-1}, C'_{i-1})$ to (R, R', C, C'), then put (R, R', C, C') in $\mathcal{F}_i$. Now, by the definition of an arc in $DAG(G)$ there is an arc from

$(R_0, R'_0, C_0, C'_0, j)$ to $(R_p, R'_p, C_p, C'_p, j+p)$ if and only if there is one $(R_{p-1}, R'_{p-1}, C_{p-1}, C'_{p-1})$ in $\mathcal{F}_{p-1}$ such that there is an ε-arc-2 from $(R_{p-1}, R'_{p-1}, C_{p-1}, C'_{p-1})$ to (R_p, R'_p, C_p, C'_p). Now, the size of each $\mathcal{F}_i$ is bounded by $n^{O(c \cdot d)}$, and since the conditions of ε-arc-1 and ε-arc-2 can be checked in time $O(n^3)$, we can check if there is an arc between two nodes in time $n^{O(c \cdot d)}$. Hence, since the number of possible arcs in $DAG(G)$ is bounded by $n^{c \cdot d} \cdot n^{c \cdot d}$, we can compute $DAG(G)$ in time $n^{O(c \cdot d)}$. □

We now prove that there is a one-to-one correspondence between the maximal paths of $DAG(G)$ and the 1-minimal **Red** (σ, ρ)-dominating sets of G. If $P = (s, v_1, v_2, \ldots, v_p, t)$ is a path in $DAG(G)$, then the *trace of* P, denoted by $\mathsf{trace}(P)$, is defined as $\{x_{j_1}, x_{j_2}, \ldots, x_{j_p}\}$ where for all $i \in \{1, 2, \ldots, p\}$, j_i is the index of the node v_i.

The following two lemmas are implied by the definition of the d-neighbor equivalence and Lemma 8.

Lemma 14. *Let $(\mu, \mu') \in \{(\sigma, \rho), (\sigma^*, \rho^*), (\sigma^-, \rho^-), (\sigma^+, \rho^+)\}$. Let also $i \in \{1, \ldots, n\}$, and $X \subseteq A_i \cap$ **Red** and $Y, Y' \subseteq \overline{A_i} \cap$ **Red**. If $Y' \equiv^d_{\overline{A_i}} Y$ then $X \cup Y$ (μ, μ')-dominates $A_i \cap$ **Blue** if and only if $X \cup Y'$ (μ, μ')-dominates $A_i \cap$ **Blue**. Symmetrically, if $X, X' \subseteq A_i \cap$ **Red** and $Y \subseteq \overline{A_i} \cap$ **Red**, and $X' \equiv^d_{A_i} X$, then $X \cup Y$ (μ, μ')-dominates $\bar{A}_i \cap$ **Blue** if and only if $X' \cup Y$ (μ, μ')-dominates $\bar{A}_i \cap$ **Blue**.*

Lemma 15. *Let $(\mu, \mu') \in \{(\sigma, \rho), (\sigma^*, \rho^*), (\sigma^-, \rho^-), (\sigma^+, \rho^+)\}$, $i \in \{1, \ldots, n\}$ and let $Z \subseteq$ **Red**. Let also $X \subseteq A_{i-1} \cap$ **Red** and $Y \subseteq \overline{A_i} \cap$ **Red**. If $X \equiv^d_{A_{i-1}} (Z \cap A_{i-1})$ and $Y \equiv^d_{\overline{A_i}} (Z \cap \overline{A_i})$, then Z (μ, μ')-dominates $\{x_i\}$ if and only if $(X \cup Y \cup (Z \cap \{x_i\}))$ (μ, μ')-dominates $\{x_i\}$.*

The next lemma shows that two maximal paths in $DAG(G)$ give rise to two different 1-minimal **Red** (σ, ρ)-dominating sets.

Lemma 16. *If there is a path $P = (s, v_1, \ldots, v_k, t)$ in $DAG(G)$, then $\mathsf{trace}(P)$ is a 1-minimal* **Red** *(σ, ρ)-dominating set of G. Moreover, $\mathsf{trace}(P) \neq \mathsf{trace}(P')$ for any other path $P' = (s, v'_1, \ldots, v'_k, t)$ in $DAG(G)$.*

Proof. Let $P = (s, v_1, \ldots, v_k, t)$ and let $D_p = \mathsf{trace}(P) \cap \{x_1, \ldots, x_p\}$ for $1 \leqslant p \leqslant n$. We will first prove by induction that for each $1 \leqslant i \leqslant k$, the set D_{j_i} satisfies the following properties (with $v_i = (R_{j_i}, R'_{j_i}, C_{j_i}, C'_{j_i}, j_i)$)

(i) $R_{j_i} \in LR^d_{j_i}$, $R'_{j_i} \in LR^{\bar{d}}_{j_i}$;

(ii) $D_{j_i} \equiv^d_{A_{j_i}} R_{j_i}$,

(iii) $D_{j_i} \cup R'_{j_i}$ (σ, ρ)-dominates $A_{j_i} \cap$ **Blue**;

(iv) Each $u \in D_{j_i}$ is either adjacent to a vertex from C'_{j_i}, or has a certificate in $A_{j_i} \cap$ **Blue**.

(v) $C_{j_i} = \mathcal{SG}_{j_i}(S_{j_i})$, where S_{j_i} is the set of vertices in $A_{j_i} \cap \mathbf{Blue}$ that are certificates and have a neighbor in $\bar{A}_{j_i} \cap \mathbf{Red}$;

By the definition of a node in $DAG(G)$, the property (i) is true for all $1 \leqslant i \leqslant k$. So, let us prove the properties (ii)-(v). By the definition of the arcs from the source node, we can easily check that the properties (ii)-(v) are all verified for $i = 1$. So, let us assume now that they are true for all $i < \ell \leqslant k$ and let us prove it for ℓ.

If there is an arc from $v_{\ell-1}$ to v_ℓ, then there should exist $(R_{j_s}, R'_{j_s}, C_{j_s}, C'_{j_s})$ for $j_{\ell-1}+1 \leqslant j_s \leqslant j_\ell-1$ such that there is an ε-arc-1 from $(R_{j_s-1}, R'_{j_s-1}, C_{j_s-1}, C'_{j_s-1})$ to $(R_{j_s}, R'_{j_s}, C_{j_s}, C'_{j_s})$ for each $j_{\ell-1} + 1 \leqslant j_s \leqslant j_\ell - 1$, and there is an ε-arc-2 from $(R_{j_\ell-1}, R'_{j_\ell-1}, C_{j_\ell-1}, C'_{j_\ell-1})$ to $(R_{j_\ell}, R'_{j_\ell}, C_{j_\ell}, C'_{j_\ell})$. By the conditions (1.1) and (2.1) we can conclude that $D_{j_\ell} \equiv^d_{A_{j_\ell}} R_{j_\ell}$ because $D_{j_{\ell-1}} \equiv^d_{A_{j_s}} R_{j_s}$ for all $j_{\ell-1}+1 \leqslant j_s \leqslant j_\ell-1$ by the condition (1.1) and $R_{j_\ell} \equiv^d_{A_{j_\ell}} (R_{j_\ell-1} \cup \{x_{j_\ell}\})$ by the condition (2.1).

Because $R'_{j_s} \equiv^d_{\bar{A}_{s-1}} R'_{j_s-1}$ for each $j_{\ell-1} + 1 \leqslant j_s \leqslant j_\ell - 1$ by (1.1) and $R'_{j_\ell-1} \equiv^d_{\bar{A}_{j_\ell-1}} R'_{j_\ell} \cup \{x_{j_\ell}\}$ by (2.1) we can conclude with inductive hypothesis, Lemmas 14 and 15, and the conditions (1.2) and (2.1) that for each $j_{\ell-1}+1 \leqslant j_s \leqslant j_\ell$ whenever $x_{j_s} \in \mathbf{Blue}$ it is (σ, ρ)-dominated by $D_{j_\ell} \cup R'_{j_\ell}$, thus proving (iii).

It remains to check (iv) and (v). For each $j_{\ell-1} + 1 \leqslant j_s \leqslant j_\ell$, the following easy facts can be derived from Lemmas 14 and 15, the definition of the d-neighbor equivalence and the fact that $D_{j_s} \cap R'_{j_s} = \varnothing$.

1. For each $v \in \bar{A}_{j_s} \cap \mathbf{Blue}$, we have $(D_{j_s} \cup R'_{j_s})$ (σ^-, ρ^-)-dominates $\{v\}$ if and only if $(R_{j_s} \cup R'_{j_s})$ (σ^-, ρ^-)-dominates $\{v\}$; and $|N(v) \cap D_{j_s}| \neq 0$ if and only if $|N(v) \cap R_{j_s}| \neq 0$.

2. For each $v \in A_{j_s-1} \cap \mathbf{Blue}$, we have $(D_{j_s} \cup R'_{j_s})$ (σ^-, ρ^-)-dominates $\{v\}$ if and only if $(D_{j_s-1} \cup R'_{j_s-1})$ (σ^-, ρ^-)-dominates $\{v\}$;

3. $(D_{j_s} \cup R'_{j_s})$ (σ^-, ρ^-)-dominates $\{x_{j_s}\}$ if and only if $(R_{j_s} \cup R'_{j_s})$ (σ^-, ρ^-)-dominates $\{x_{j_s}\}$.

4. Each $u \in D_{j_s}$ distinct from x_{j_ℓ} either has a certificate in $A_{j_s} \cap \mathbf{Blue}$ *w.r.t.* D_{j_s} or is adjacent to a vertex from C'_{j_s}. Indeed, u has by induction either a certificate v from $A_{j_s-1} \cap \mathbf{Blue}$ *w.r.t.* D_{j_s-1}, or is adjacent to a vertex in C'_{j_s-1}. If u has a certificate v from $A_{j_s-1} \cap \mathbf{Blue}$ *w.r.t.* D_{j_s-1}, then by the fact 2 and Lemma 9 the vertex v is still a certificate for u *w.r.t.* D_{j_s}. If u is adjacent to some vertex in C'_{j_s-1}, then by induction and the conditions (1.2) and (2.2) either it is adjacent to some vertex in C'_{j_s} or it is adjacent to x_{j_s} which is (σ^-, ρ^-)-dominated by $D_{j_s} \cup R'_{j_s}$ following the fact 3.

From the facts 1 and 2 we can conclude that $(D_{j_s} \cup R'_{j_s})$ (σ^-, ρ^-)-dominates C_{j_s} for all $j_{\ell-1} + 1 \leqslant j_s \leqslant j_\ell$. Hence, $(D_{j_\ell} \cup R'_{j_\ell})$ (σ^-, ρ^-)-dominates C_{j_ℓ}. Moreover, from the fact 4 we know that each $u \in D_{j_{\ell-1}}$ is either adjacent to a vertex in C'_{j_ℓ} or has a certificate *w.r.t.* D_{j_ℓ} in $A_{j_\ell} \cap \mathbf{Blue}$. In order to prove that (iv) is satisfied it remains then to check that x_{j_ℓ} has a certificate in $A_{j_\ell} \cap \mathbf{Blue}$ or has a neighbor in C'_{j_ℓ}. But this is guaranteed with the existence of the arc $v_{\ell-1}$ to v_ℓ by the conditions (1.2.a), (2.2.a), (2.3), and the properties (iv)-(v) by inductive hypothesis.

In order to check the condition (v) it is sufficient to notice that whenever x_{j_s} is (σ^-, ρ^-)-dominated by $(D_{j_s} \cup R'_{j_s})$ for each $j_{\ell-1} + 1 \leqslant j_s \leqslant j_\ell$, by the conditions (1.2.a) and (2.2.a) $C_{j_s} = \mathcal{SG}_{j_s}(C_{j_s-1} \cup \{x_{j_s}\})$. Thus, by inductive hypothesis, the fact 2 and Lemma 9 we can conclude that C_{j_s} is exactly $\mathcal{SG}_{j_s}(S_{j_s})$ where S_{j_s} is the set of vertices in $A_{j_s} \cap \mathbf{Blue}$ that are certificates and have a neighbor in $\bar{A}_{j_s} \cap \mathbf{Red}$.

To end the proof we need to prove that whenever $\mathsf{trace}(P) = \mathsf{trace}(P')$ for any other path P' from the source node to the terminal node, then $P = P'$. For that we prove by induction that $C'_{j_i} = \mathcal{GG}_{j_i}(S'_{j_i})$ where S'_{j_i} is the set of vertices in $\bar{A}_{j_i} \cap \mathbf{Blue}$ that are (σ^-, ρ^-)-dominated by $D_{j_i} \cup R'_{j_i}$ and have a neighbor in $A_{j_i} \cap \mathbf{Red}$. By the condition (T2) in the definition of an arc to the terminal node this is satisfied by C'_{j_k}. So, if we assume that $C'_{j_i} = \mathcal{GG}_{j_i}(S'_{j_i})$ for all $\ell < i \leqslant k$, then as for the condition (v) the inductive hypothesis, the fact 2 and Lemma 9 guarantee that C'_{j_ℓ} is exactly $\mathcal{GG}_{j_\ell}(S'_{j_\ell})$.

So now for each i the sets C_{j_i} and C'_{j_i} are uniquely determined by $\mathsf{trace}(P)$, which means that whenever $\mathsf{trace}(P) = \mathsf{trace}(P')$ because $\equiv^d_A$ is an equivalence relation we should conclude that $P = P'$. □

The following lemma tells that to every 1-minimal **Red** (σ, ρ)-dominating set corresponds a maximal path in $DAG(G)$.

Lemma 17. *If G has a 1-minimal* **Red** *(σ, ρ)-dominating set D, then there is a path $P = (s, v_1, v_2, \ldots, v_k, t)$ in $DAG(G)$ such that $D = \mathsf{trace}(P)$.*

Proof. Let $D = \{x_{j_1}, \ldots, x_{j_k}\}$ such that $j_1 < j_2 < \cdots < j_k$. For each $i \in \{1, \ldots, k\}$, we let $D_{j_i} = \{x_{j_1}, \ldots, x_{j_i}\}$. Let also $R_{j_i} \in LR^d_{j_i}$ be such that $R_{j_i} \equiv^d_{A_{j_i}} D_{j_i}$, and let $R'_{j_i} \in LR^d_{j_i}$ be such that $R'_{j_i} \equiv^d_{\bar{A}_{j_i}} (D \cap \bar{A}_{j_i})$. For each i we let $C_{j_i} = \mathcal{SG}_{j_i}(S_{j_i})$ where S_{j_i} is the set of vertices in $A_{j_i} \cap \mathbf{Blue}$ that are certificates and have a neighbor in $\bar{A}_{j_i} \cap \mathbf{Red}$ and similarly let $C'_{j_i} = \mathcal{GG}_{j_i}(S'_{j_i})$ where S'_{j_i} is the set of vertices in $\bar{A}_{j_i} \cap \mathbf{Blue}$ that are certificates and have a neighbor in $A_{j_i} \cap \mathbf{Red}$. Hence, $v_i = (R_{j_i}, R'_{j_i}, C_{j_i}, C'_{j_i}, j_i)$ is a node of $DAG(G)$ for each $1 \leqslant i \leqslant k$.

We first observe that there is an arc from the source node s to v_1. Indeed, by the definition of R_{j_i}, R'_{j_i}, we have that $\{x_{j_1}\} \equiv^d_{A_{j_1}} R_{j_1}$ and $\{x_{j_1}\} \cup R'_{j_1}$ (σ, ρ)-dominates $A_{j_1} \cap \mathbf{Blue}$, and by the choices of C_{j_i} and C'_{j_i} the condition (S2) is satisfied and since x_{j_1} has a certificate *w.r.t.* D, the condition (S3)

is satisfied. For similar reasons one can prove that there is an arc from v_k to the terminal node t.

We now claim that there is an arc from v_i to v_{i+1} for $1 \leqslant i < k$. For each $j_i < j_s < j_{i+1}$, we let $(R_{j_s}, R'_{j_s}, C_{j_s}, C'_{j_s})$ be such that $R_{j_s} \equiv^d_{A_{j_s}} R_{j_s-1}$, $R'_{j_s} \equiv^d_{A_{j_s-1}} R'_{j_s-1}$, and $R_{j_{i+1}} \equiv^d_{A_{j_{i+1}}} R_{j_{i+1}-1} \cup \{x_{j_{i+1}}\}$ and $R'_{j_{i+1}-1} \equiv^d_{A_{j_{i+1}-1}} R'_{j_{i+1}} \cup \{x_{j_{i+1}}\}$. It is straightforward to prove by induction on $j_{i+1} - j_i$ that there exists an ε-arc-1 from $(R_{j_s-1}, R'_{j_s-1}, C_{j_s-1}, C'_{j_s-1})$ to $(R_{j_s}, R'_{j_s}, C_{j_s}, C'_{j_s})$ for each $j_i < j_s < j_{i+1}$ and there is an ε-arc-2 from $(R_{j_{i+1}-1}, R'_{j_{i+1}-1}, C_{j_{i+1}-1}, C'_{j_{i+1}-1})$ to $(R_{j_{i+1}}, R'_{j_{i+1}}, C_{j_{i+1}}, C'_{j_{i+1}})$. □

By Lemmas 16 and 17 we can state the following.

Proposition 18. *Let $\mathcal{P}$ be the set of paths in $DAG(G)$ from the source node to the terminal node. The mapping which associates with every $P \in \mathcal{P}$ $\mathsf{trace}(P)$ is a one-to-one correspondence with the set of 1-minimal* **Red** *(σ, ρ)-dominating sets.*

By Proposition 18 it suffices to count and enumerate the traces of the maximal paths in $DAG(G)$. We will now explain how to count and then use the counting to enumerate the traces of these paths in $DAG(G)$. We start from a topological ordering of $DAG(G)$, say $s = v_1, v_2, \ldots, v_m = t$. Since $DAG(G)$ is a DAG, any arc is of the form (v_i, v_j) with $i < j$. The counting will follow this topological ordering. We initially set $Np(v) = -1$ for all nodes $v \neq t$ and we set $Np(v_m) = 1$. For each $j < m$ we let

$$Np(v_j) = \sum_{\substack{(v_j, v_\ell) \in E(DAG(G)) \\ Np(v_\ell) \neq -1}} Np(v_\ell).$$

Fact 19. *One can compute the values of $Np(v_j)$ for all $j \in \{1, 2, \ldots, m\}$ in time $n^{O(c \cdot d)}$.*

Proof. By induction on j. By definition $Np(v)$ can be computed in time $O(1)$ for all v that have exactly one outgoing arc, which enters t. For every $j < m$, in order to compute $Np(v_j)$ we first set a counter **nb** to 0, and add $Np(v_\ell)$ to **nb** whenever $(v_j, v_\ell) \in E(DAG(G))$ and $Np(v_\ell) \neq -1$. We finally set $Np(v_j)$ to **nb**. The correctness of the computation of $Np(v_j)$ follows from the definition. Since the degree of a node is bounded by $n^{O(c \cdot d)}$, we can update **nb** in time $n^{O(c \cdot d)}$. Now since the number of nodes and of arcs is bounded by $n^{O(c \cdot d)}$, we obtain the claimed running time. □

For $1 \leqslant j \leqslant m$, we let $\mathcal{S}_j = \{P \mid P \text{ is a path starting at } v_j \text{ and ending at } t\}$. One can prove easily by induction the following.

Lemma 20. $\mathcal{S}_j = \biguplus_{\substack{(v_j, v_\ell) \in E(DAG(G)) \\ Np(v_\ell) \neq -1}} \{v_j + P \mid P \in \mathcal{S}_\ell\}$ *for each $1 \leqslant j \leqslant m$.*

The following follows directly from the definition of $Np(v_j)$ and Lemma 20.

Lemma 21. *$|\mathcal{S}_j| = Np(v_j)$ for each $1 \leqslant j \leqslant m$.*

Theorem 22. *One can count the number of 1-minimal* **Red** *(σ, ρ)-dominating sets of a given graph G in time $n^{O(c \cdot d)}$.*

Proof. We first construct the DAG $DAG(G)$ and by Lemma 13 this can be done in time $n^{O(c \cdot d)}$. By Proposition 18 the mapping which associates with every path $P \in \mathcal{S}_1$ its trace $\mathsf{trace}(P)$ is a one-to-one correspondence between $\mathcal{S}_1$ and all the 1-minimal **Red** (σ, ρ)-dominating set in G. So, it is enough to determine the size of $\mathcal{S}_1$. By Lemma 21, $|\mathcal{S}_1| = Np(s)$ and since by Fact 19 we can compute in time $n^{O(c \cdot d)}$ all the values $Np(v_j)$ for all $1 \leqslant j \leqslant m$, we conclude that one can compute in time $n^{O(c \cdot d)}$ the number of 1-minimal **Red** (σ, ρ)-dominating sets in G. □

We now turn to the enumeration of the 1-minimal **Red** (σ, ρ)-dominating sets. For each node v in $DAG(G)$ of index j we denote by $\mathsf{vert}(v)$ the vertex x_j of G. The algorithm is depicted in Figures 1 and 2. The algorithm consists in enumerating the paths in $\mathcal{S}_1$ in a Depth-First Search manner.

Algorithm EnumMinDom($DAG(G)$)
1. Remove all nodes v such that $Np(v) = -1$ or v is not reachable from the source v_1
2. **for** each $(s, v_i) \in E(DAG(G))$
3. EnumPath($DAG(G), \{\mathsf{vert}(v_i)\}, v_i$)
4. **end for**

Figure 1: The enumeration of 1-minimal **Red** (σ, ρ)-dominating sets

Algorithm EnumPath($DAG(G), S \subseteq V(G), v_i$)
1. **if** $v_i = t$, then **output** S **and stop**
2. **for** each $(v_i, v_j) \in E(DAG(G))$
3. EnumPath($DAG(G), S \cup \{\mathsf{vert}(v_j)\}, v_j$)
4. **end for**

Figure 2: The enumeration of $\mathcal{S}_i$

Theorem 23. *We can enumerate all the 1-minimal* **Red** *(σ, ρ)-dominating sets of a given graph G with linear delay and with polynomial space.*

Proof. First notice that after removing all the nodes v such that $Np(v) = -1$ or v is not reachable from the source node, every remaining node is in a path from the source node to the terminal node. Now, it is easy to prove by induction using Lemmas 20 and 21 that the algorithm $\mathsf{EnumPath}(DAG(G), S, v_i)$ uses $n^{O(c \cdot d)}$ space and enumerates the set $\{S \cup P \mid P \in \mathcal{S}_i\}$, the delay between two consecutive outputs P_1 and P_2 bounded by $O(|P_2 \backslash P_1|)$. In fact if before calling $\mathsf{EnumPath}$ we order the out-neighbors of each node following their distances to the terminal node and uses this ordering in the recursive calls we guarantee that the time between the output of P and the next output Q is bounded by $O(|Q|)$. Therefore, the algorithm $\mathsf{EnumMinDom}(DAG(G))$ enumerates, with same delay as $\mathsf{EnumPath}$ the set of 1-minimal **Red** (σ, ρ)-dominating sets and uses $n^{O(c \cdot d)}$ space. □

3.3 Maximal sets

We now explain how to construct the DAG $DAGM(G)$ so that the maximal paths from the source node to the terminal node corresponds to the 1-maximal **Red** (σ, ρ)-dominating sets, and conversely each 1-maximal **Red** (σ, ρ)-dominating set corresponds to such a path.

The nodes of $DAG(G)$ and of $DAGM(G)$ are the same, and the two DAGs only differ in the conditions for adding arcs. In the case of 1-minimal **Red** (σ, ρ)-dominating sets whenever a vertex x_j is considered to be in the solution it must have a certificate, while it must have a certificate when considering 1-maximal **Red** (σ, ρ)-dominating sets only if not included in the solution. In both cases, having a certificate means : either it does have a certificate in $A_j \cap \mathbf{Blue}$ or it is adjacent to a vertex in C'. The checking of these conditions are possible with the help of Lemma 9.

The nodes of $DAGM(G)$.

$$(R, R', C, C', i) \in LR_i^d \times LR_{\bar{i}}^d \times 2^{A_i \cap \mathbf{Blue}} \times 2^{\bar{A}_i \cap \mathbf{Blue}} \times \{1, \ldots, n\}$$

is a node of $DAGM(G)$ whenever $x_i \in \mathbf{Red}$, $C = \mathcal{SG}_i(C)$ and $C' = \mathcal{GG}_i(C')$. We call i the *index* of (R, R', C, C', i). Finally $s = (\varnothing, \varnothing, \varnothing, \varnothing, 0)$ is the *source node* and $t = (\varnothing, \varnothing, \varnothing, \varnothing, n+1)$ is the *terminal node* of $DAG(G)$.

The arcs of $DAGM(G)$. There is an arc from $(R_0, R'_0, C_0, C'_0, j)$ to $(R_p, R'_p, C_p, C'_p, j+p)$ with $j < j+p \leqslant n$ if there exist (R_1, R'_1, C_1, C'_1), $\ldots$, $(R_{p-1}, R'_{p-1}, C_{p-1}, C'_{p-1})$ such that

$$(R_i, R'_i, C_i, C'_i) \in LR_{j+i}^d \times LR_{\overline{j+i}}^d \times 2^{A_{j+i} \cap \mathbf{Blue}} \times 2^{\bar{A}_{j+i} \cap \mathbf{Blue}}$$

for all $1 \leqslant i \leqslant p-1$, and

(A1) for each $0 \leqslant i \leqslant p-2$,

(1.1) $R_i \equiv^d_{A_{j+i+1}} R_{i+1}$ and $R'_i \equiv^d_{\bar{A}_{j+i=1}} R'_{i+1}$, and

(1.2) if ($x_{j+i+1} \notin$ **Blue** or ($x_{j+i+1} \in$ **Blue** and $|N(x_{j+i+1}) \cap (R_i \cup R'_{i+1})| \in \rho$ and $|N(x_{j+i+1}) \cap (R_i \cup R'_{i+1})| \notin \rho^+$)) then ($C_{i+1} = \mathcal{SG}_{j+i+1}(C_i)$ and $C'_i = \mathcal{GG}_{j+i}(C'_{i+1})$), otherwise we should have ($|N(x_{j+i+1}) \cap (R_i \cup R'_{i+1})| \in \rho^+$) and

(1.2.a) if $N(x_{j+i+1}) \cap (\bar{A}_{j+i+1} \cap$ **Red**$) \neq \emptyset$, then $C_{i+1} = \mathcal{SG}_{j+i+1}(C_i \cup \{x_{j+i+1}\})$, else $C_{i+1} = \mathcal{SG}_{j+i+1}(C_i)$, and

(1.2.b) if $N(x_{j+i+1}) \cap (A_{j+i} \cap$ **Red**$) \neq \emptyset$, then $C'_i = \mathcal{GG}_{j+i}(C'_{i+1} \cup \{x_{j+i+1}\})$, else $C'_i = \mathcal{GG}_{j+i}(C'_{j+i+1})$.

(1.3) if $x_{j+i+1} \in$ **Red**, then either ($N(x_{j+i+1}) \cap (C_i \cup C'_{i+1}) \neq \emptyset$) or (($x_{j+i+1} \in$ **Blue** and $|N(x_{j+i+1}) \cap (R_i \cup R'_{i+1})| \in \rho^*$).

(A2) $R_p \equiv^d_{A_{j+p}} (R_{p-1} \cup \{x_{j+p}\})$, $R'_{p-1} \equiv^d_{\bar{A}_{j+p-1}} (R'_p \cup \{x_{j+p}\})$, $x_{j+p} \in$ **Red**, and

(2.1) if $x_{j+p} \in$ **Blue**, then $|N(x_{j+p}) \cap (R_{p-1} \cup R'_p)| \in \sigma$,

(2.2) if ($x_{j+p} \notin$ **Blue** or ($x_{j+p} \in$ **Blue** and $|N(x_{j+p}) \cap (R_{p-1} \cup R'_p)| \notin \sigma^+$)), then ($C_p = \mathcal{SG}_{j+p}(C_{p-1})$ and $C'_{p-1} = \mathcal{GG}_{j+p-1}(C'_p)$), otherwise we should have ($|N(x_{j+p}) \cap (R_{p-1} \cup R'_p)| \in \sigma^+$) and

(2.2.a) if $N(x_{j+p}) \cap (\bar{A}_p \cap$ **Red**$) \neq \emptyset$, then $C_p = \mathcal{SG}_{j+p}(C_{p-1} \cup \{x_{j+p}\})$, else $C_p = \mathcal{SG}_{j+p}(C_{p-1})$, and

(2.2.b) if $N(x_{j+p}) \cap (A_{p-1} \cap$ **Red**$) \neq \emptyset$, then $C'_{p-1} = \mathcal{GG}_{j+p-1}(C'_p \cup \{x_{j+p}\})$, else $C'_{j+p-1} = \mathcal{GG}_{j+p-1}(C'_p)$.

We now define arcs from the source node. For a node $u = (R, R', C, C', j)$, let $S_j = \{x \in (A_j \cap$ **Blue**$)\backslash\{x_j\} \mid N(x) \cap (\bar{A}_j \cap$ **Red**$) \neq \emptyset$ and $|N(x) \cap (\{x_j\} \cup R')| \in \rho^+\}$. There is an arc from the source node to u if

(S1) $\{x_j\} \equiv^d_{A_j} R$ and $(\{x_j\} \cup R')$ (σ, ρ)-dominates $A_j \cap$ **Blue**,

(S2) if ($x_j \in$ **Blue** and $|N(x_j) \cap R'| \in \sigma^+$) then $C = \mathcal{SG}_j(S_j \cup \{x_j\})$, otherwise $C = \mathcal{SG}_j(S_j)$, and

(S3) for each red vertex $x \in A_j\backslash\{x_j\}$, either ($N(x) \cap (C' \cup C) \neq \emptyset$) or ($x \in$ **Blue** and $|N(x) \cap (R' \cup R)| \in \rho^*$).

We finally define the arcs to the terminal node. There is an arc from a node (R, R', C, C', j) to the terminal node if

(T1) $|N(x) \cap (R \cup R')| \in \rho$ for each $x \in \bar{A}_{j+1} \cap$ **Blue**, and

(T2) $C' = \mathcal{GG}_j(\{x \in \bar{A}_j \cap$ **Blue** $\mid N(x) \cap (A_j \cap$ **Red**$) \neq \emptyset$ and $|N(x) \cap R| \in \rho^+\})$,

(T3) for each red vertex in $\bar{A}_j$, then $N(x) \cap (C \cup C') \neq \varnothing$ or ($|N(x) \cap (R \cup R')| \in \rho^*$ if $x \in$ **Blue**).

$DAGM(G)$ is clearly a DAG, and as for $DAG(G)$ one can construct it in time $n^{O(c \cdot d)}$. Similarly, one can observe that if $P := (s, v_1, \ldots, v_k, t)$ is a maximal path from the source node to the terminal node, then if we let $D_{j_i} := \{x_{j_1}, \ldots, x_{j_i}\}$ with $v_i := (R_{j_i}, R'_{j_i}, C_{j_i}, C'_{j_i}, j_i)$, then

(i) $R_{j_i} \in LR^d_{j_i}$, $R'_{j_i} \in LR^d_{\bar{j}_i}$;

(ii) $D_{j_i} \equiv^d_{A_{j_i}} R_{j_i}$,

(iii) $D_{j_i} \cup R'_{j_i}$ (σ, ρ)-dominates $A_{j_i} \cap$ **Blue**;

(iv) Each $u \in A_{j_i} \backslash D_{j_i}$ is either adjacent to a vertex from C'_{j_i}, or has a certificate in $A_{j_i} \cap$ **Blue**.

(v) $C_{j_i} = \mathcal{SG}_{j_i}(S_{j_i})$, where S_{j_i} is the set of vertices in $A_{j_i} \cap$ **Blue** that are certificates and have a neighbor in $\bar{A}_{j_i} \cap$ **Red**;

(vi) $C'_{j_i} = \mathcal{GG}_{j_i}(S'_{j_i})$ where S'_{j_i} is the set of vertices in $\bar{A}_{j_i} \cap$ **Blue** that are certificates and have a neighbor in $A_{j_i} \cap$ **Red**.

Hence, we can prove counterparts to Lemmas 16 and 17 and deduce the following theorem from Section 3.2.

Theorem 24. *The set of* 1*-maximal* **Red** (σ, ρ)*-dominating sets in* G *can be enumerated with linear delay and with polynomial space. We can moreover count in time* $n^{O(c \cdot d)}$ *the number of* 1*-maximal* **Red** (σ, ρ)*-dominating sets in* G*.*

4 Enumeration of minimal dominating sets for unit square graphs

In this section we prove that all minimal dominating sets of a unit square graph can be enumerated in incremental polynomial time. In Section 4.1 we show that the class of unit square graphs has locally bounded LMIM-width. In Section 4.2 we use this property and Theorem 4, to obtain an enumeration algorithm for minimal dominating sets. To do it, we use the *flipping* method proposed by Golovach, Heggernes, Kratsch and Villanger in [13].

4.1 Local LMIM-width of unit square graphs

First, we introduce some additional notations. For $x, y \in \mathbb{R}$ such that $x \leqslant y$, $[x, y] = \{z \in \mathbb{R} \mid x \leqslant z \leqslant y\}$. Let G be a unit square graph and suppose that $f\colon V(G) \to \mathbb{Q}^2$ is a realization of G. (See Section 2 for more details on the point model of unit square graphs used in our paper.) For a vertex $v \in V(G)$, $\mathbf{frac}(v) = x_f(v) - \lfloor x_f(v) \rfloor$ is the fractional part of the x-coordinate of the point representing v.

Lemma 25. *Let G be a unit square graph with a realization f such that for every $v \in V(G)$ the point $f(v)$ belongs to $[1, w] \times [1, h]$, where $h, w \in \mathbb{N}$. If for $x \in [1, w]$, $A = \{v \in V(G) \mid x_f(v) = x\}$ is a non-empty proper subset of $V(G)$, then $\mathbf{mim}_G(A, \overline{A}) \leqslant h$.*

Proof. Let M be a maximum induced matching in $G[A, \bar{A}]$. Denote by M_A the set of end-vertices of the edges of M in A and let M_B be the set of end-vertices of the matching in $\bar{A}$. As all the vertices of M_A have the same x-coordinate, and each vertex of M_B is adjacent to some vertex in M_A, a vertex $u \in M_A$ is adjacent to a vertex $v \in M_B$ only if $|y_f(u) - y_f(v)| < 1$. Denote by $a_1, \ldots, a_k$ and $b_1, \ldots, b_k$ the vertices of M_A and M_B respectively and assume that they are ordered by the increase of their y-coordinates.

We claim that $a_i b_i \in M$ for $i \in \{1, \ldots, k\}$. To obtain a contradiction, suppose that there is a_i that is not adjacent to b_i and choose the minimum index i for which it holds. Then $a_i b_j \in M$ and $b_i a_s \in M$ for some $j, s > i$. If b_i is adjacent to a_s but not a_i, we must have $|y_f(b_i) - y_f(a_i)| \geqslant 1$ and $|y_f(b_i) - y_f(a_s)| < 1$. Since $y_f(a_s) \geqslant y_f(a_i)$, $y_f(b_i) \geqslant y_f(a_i) + 1$. But as $y_f(b_j) \geqslant y_f(b_i) \geqslant y_f(a_i) + 1$, b_j and a_i cannot be adjacent after all; a contradiction.

Now we show that $y_f(a_i) \geqslant y_f(b_{i-1}) + 1$ and $y_f(b_i) \geqslant y_f(a_{i-1}) + 1$ for $i \in \{2, \ldots, k\}$. Because $a_{i-1} b_{i-1} \in M$, $|y_f(a_{i-1}) - y_f(b_{i-1})| < 1$. As $a_{i-1} b_i, a_i b_{i-1} \notin E(G)$, $|y_f(a_{i-1}) - y_f(b_i)| \geqslant 1$ and $|y_f(a_i) - y_f(b_{i-1})| \geqslant 1$. We have that $y_f(a_i) \geqslant y_f(b_{i-1}) + 1$ and $y_f(b_i) \geqslant y_f(a_{i-1}) + 1$, because $y_f(a_i) \geqslant y_f(a_{i-1})$ and $y_f(b_i) \geqslant y_f(b_{i-1})$.

Next, we claim that $y_f(a_i), y_f(b_i) \geqslant i$. Clearly, $y_f(a_1), y_f(b_1) \geqslant 1$. Because $y_f(a_i) \geqslant y_f(b_{i-1}) + 1$ and $y_f(b_i) \geqslant y_f(a_{i-1}) + 1$ for $i \in \{2, \ldots, k\}$, we have that the claim holds for all $i \in \{1, \ldots, k\}$ by induction.

Because $k \leqslant y_f(a_k) \leqslant h$, we conclude that $\mathbf{mim}_G(A, \bar{A}) = k \leqslant h$. □

Lemma 26. *Let G be a unit square graph with a realization f such that for every $v \in V(G)$ the point $f(v)$ belongs to $[1, w] \times [1, h]$, where $h, w \in \mathbb{N}$. Then $\mathbf{lmimw}(G) \leqslant 2hw$. Moreover, a linear ordering of the vertices of MIM-width at most $2hw$ can be constructed in polynomial time.*

Proof. Let $v_1, \ldots, v_n$ be the vertices of G ordered by increasing **frac**-value, i.e., $\mathbf{frac}(v_i) \leqslant \mathbf{frac}(v_j)$ if $i \leqslant j$. We show that this is an ordering of MIM-width at most $2hw$.

For contradiction, assume that for some $i \in \{1, \ldots, n-1\}$, $\mathbf{mim}_G(A, \bar{A}) \geqslant 2hw + 1$ where $A = \{v_1, \ldots, v_i\}$, i.e., the graph $G[A, \bar{A}]$ has an induced matching M of size $2hw + 1$. Let $V(M)$ denote the set of the end-vertices of the edges of M. By the pigeonhole principle, for some positive integer $p \leqslant w$ there are at least $2h + 1$ vertices $v \in V(M) \cap A$ so that $\lfloor x_f(v) \rfloor = p$. Denote this subset of $A \cap V(M)$ by C_A, and denote by C_B the vertices of $V(M) \backslash A$ adjacent to C_A. Let $t = \max\{\mathbf{frac}(v) \mid v \in A\}$, and observe that $\{v \mid \mathbf{frac}(v) < t\} \subset A$ and $\{v \mid \mathbf{frac}(v) > t\} \cap A = \varnothing$. We now partition C_A into two parts $C_A^{<t} = \{v \in C_A \mid \mathbf{frac}(v) < t\}$ and $C_A^{=t} = \{v \in C_A \mid \mathbf{frac}(v) = t\}$ and argue that neither of these parts can be of size more than h, contradicting that $|C_A| \geqslant 2h + 1$.

We first show that $|C_A^{=t}| \leqslant h$. For each $v \in C_A^{=t}$, $\lfloor x_f(v) \rfloor = p$ and $\mathbf{frac}(v) = t$. Hence, $x_f(v) = p + t$ for all $v \in C_A^{=t}$. By Lemma 25, the size of the maximum induced matching in $G[C_A^{=t}, C_B]$ is at most h and this implies that $|C_A^{=t}| \leqslant h$.

To show that also $|C_A^{<t}| \leqslant h$, we will show that for the sake of the induced matching, all the x-coordinates of the vertices v of $C_A^{<t}$ might as well have $\mathbf{frac}(v) = 0$, and therefore we can apply Lemma 25.

Let $v \in C_A^{<t}$. Then we construct a new vertex v' represented by the point $(\lfloor x_f(v) \rfloor, y_f(v))$. We will now show that v' is adjacent to a vertex $u \in C_B$ if and only if v is adjacent to u. As the y-coordinates of v and v' are the same, we only need to prove that $|x_f(v) - x_f(u)| < 1$ if and only if $|\lfloor x_f(v) \rfloor - x_f(u)| < 1$.

Suppose that v is adjacent to u but v' is not. Because $x_f(v) \geqslant \lfloor x_f(v) \rfloor$, we have that $x_f(v) + 1 \geqslant x_f(u) > \lfloor x_f(v) \rfloor + 1$. However, that means $\mathbf{frac}(u) \leqslant \mathbf{frac}(v) < t$, which implies that $u \in A$ contradicting $u \in C_B$. Similarly, suppose v' is adjacent to u but v is not. Now $\lfloor x_f(v) \rfloor - 1 < x_f(u) \leqslant x_f(v) - 1$, which again implies that $\mathbf{frac}(u) \leqslant \mathbf{frac}(v) < t$, contradicting that u is in C_B.

Consider $S = \{v' \mid v \in C_A^{<t}\}$, where each v' is represented by the point $(\lfloor x_f(v) \rfloor, y_f(v)) = (p, y_f(v))$. Because each $v' \in S$ is adjacent to $u \in C_B$ if and only if v is adjacent to u, by Lemma 25, $|C_A^{<t}| = |S| \leqslant h$.

It remains to show that the ordering of $V(G)$ can be constructed in polynomial time. Clearly, the ordering can be done in time $O(n \log n)$ if we assume that we can compute $\mathbf{frac}(v)$ and compare the $\mathbf{frac}$-values of two vertices in time $O(1)$. Otherwise, if the table of the values $f \colon V(G) \to \mathbb{Q}^2$ is given in the input, we still can produce the ordering in polynomial time. □

Now we are ready to show that the class of unit square graphs has locally bounded LMIM-width.

Theorem 27. *For a unit square graph G, $u \in V(G)$ and a positive integer r, $\mathbf{lmimw}(G[N_G^r[u]]) = O(r^2)$. Moreover, if a realization $f \colon V(G) \to \mathbb{Q}^2$ of G is given, then a linear ordering of the vertices of MIM-width $O(r^2)$ can be constructed in polynomial time.*

Proof. Let f be a realization of G. Without loss of generality we may assume that $\min\{x_f(v) \mid v \in N_G^r[u]\} = \min\{y_f(v) \mid v \in N_G^r[u]\} = 1$. Otherwise, we can shift the points representing vertices. For any $v \in N_G^r[u]$, $|x_f(u) - x_f(v)| \leqslant r$ and $|y_f(u) - y_f(v)| \leqslant r$. We obtain that $f(v) \in [1, 2r+1] \times [1, 2r+1]$. By Lemma 26, $\mathbf{lmimw}(G[N_G^r[u]]) = O(r^2)$ and the corresponding ordering of the vertices can be constructed in polynomial time. □

4.2 Enumeration by flipping for graphs of locally bounded LMIM-width

We use a variant of the *flipping* method proposed by Golovach, Heggernes, Kratsch and Villanger in [13]. Given a minimal dominating set D^*, the flipping operation replaces an isolated vertex of $G[D^*]$ with its neighbor outside of D^*, and, if necessary, adds or deletes some vertices to obtain new minimal dominating sets D, such that $G[D]$ has more edges compared to $G[D^*]$. The enumeration algorithm starts with enumerating all maximal independent sets of the input graph G using the algorithm of Johnson, Papadimitriou, and Yannakakis [15], which gives the initial minimal dominating sets. Then the flipping operation is applied to every appropriate minimal dominating set found, to find new minimal dominating sets inducing subgraphs with more edges.

Let G be a graph. Let also $D \subseteq V(G)$. For $u \in D$,

$$C_D[u] = \{v \in V(G) \mid v \in N_G[u] \backslash N_G[D \backslash \{v\}]\}$$

and

$$C_D(u) = \{v \in V(G) \mid v \in N_G(u) \backslash N_G[D \backslash \{v\}]\} = C_D[u] \backslash \{u\}.$$

Observe that if D is a minimal dominating set, then $C_D[u]$ is the set of certificates for a vertex $u \in D$.

Let us describe the variant of the flipping operation from [13], that we use. Let G be the input graph; we fix an (arbitrary) order of its vertices: $v_1, \ldots, v_n$. Suppose that D' is a dominating set of G. We say that the minimal dominating set D is obtained from D' by *greedy removal of vertices* (*with respect to order* $v_1, \ldots, v_n$) if we initially let $D = D'$, and then recursively apply the following rule: *If D is not minimal, then find a vertex v_i with the smallest index i such that $D \backslash \{v_i\}$ is a dominating set in G, and set $D = D \backslash \{v_i\}$.* Clearly, when we apply this rule, we never remove vertices of D' that have certificates. Whenever greedy removal of vertices of a dominating set is performed, it is done with respect to this ordering.

Let D be a minimal dominating set of G such that $G[D]$ has at least one edge uw. Then the vertex $u \in D$ is dominated by the vertex $w \in D$. Therefore, $C_D[u] = C_D(u) \neq \varnothing$. Let X be a non-empty inclusion maximal set of pairwise non-adjacent vertices in $C_D(u)$. Consider the set $D' = (D \backslash \{u\}) \cup X$. Notice that D' is a dominating set in G, since all vertices

of $C_D(u)$ are dominated by X by the maximality of X and u is dominated by w, but D' is not necessarily minimal, because it can happen that X dominates all the certificates of some vertex of $D\backslash\{u\}$. We apply greedy removal of vertices to D' to obtain a minimal dominating set. Let Z be the set of vertices that are removed by this to ensure minimality. Observe that $X \cap Z = \varnothing$ and $u \notin Z$ by the definition of these sets; in fact there is no edge between a vertex of X and a vertex of Z. Finally, let $D^* = ((D\backslash\{u\}) \cup X)\backslash Z$.

It is important to notice that $|E(G[D^*])| < |E(G[D])|$. Indeed, to construct D^*, we remove the endpoint u of the edge $uw \in E(G[D])$ and, therefore, reduce the number of edges. Then we add X but these vertices form an independent set in G and, because they are certificates for u with respect to D, they are not adjacent to any vertex of $D\backslash\{u\}$. Therefore, $|E(G[D^*])| \leqslant |E(G[D'])| < |E(G[D])|$.

The *flipping* operation is exactly the *reverse* of how we generated D^* from D; *i.e.*, it replaces a non-empty independent set X in $G[D^*]$ such that $X \subseteq G[D^*] \cap N_G(u)$ for a vertex $u \notin D^*$ with their neighbor u in G to obtain D. In particular, we are interested in all minimal dominating sets D that can be generated from D^* in this way. Given D and D^* as defined above, we say that D^* is a *parent of D with respect to flipping u and X*. It is important to note that each minimal dominating set D such that $E(G[D]) \neq \varnothing$ has a unique parent with respect to flipping of any $u \in D \cap N_G[D\backslash\{u\}]$ and a maximal independent set $X \subseteq C_D(u)$, as Z is lexicographically selected by a greedy algorithm. Similarly, we say that D is a *child* of D^* (with respect to flipping u and X) if D^* is the parent of D (with respect to flipping u and X).

The proof of the following lemma is implicit in [13].

Lemma 28 ([13])**.** *Suppose that for a graph G, all independent sets $X \subseteq N_G(u)$ for a vertex u can be enumerated in polynomial time. Suppose also that there is an enumeration algorithm $\mathcal{A}$ that, given a minimal dominating set D^* of a graph G such that $G[D^*]$ has an isolated vertex, a vertex $u \in V(G)\backslash D^*$ and a non-empty independent set X of $G[D^*]$ such that $X \subseteq D^* \cap N_G(u)$, generates with polynomial delay a family of minimal dominating sets $\mathcal{D}$ with the property that $\mathcal{D}$ contains all minimal dominating sets D that are children of D^* with respect to flipping u and X. Then all minimal dominating sets of G can be enumerated in incremental polynomial time.*

To obtain our main result, we will show that there is indeed an algorithm as algorithm $\mathcal{A}$ described in the statement of Lemma 28 when the input graph G is a unit square graph. We show that we can construct $\mathcal{A}$ by reduction to the enumeration of minimal **Red** dominating sets in an auxiliary colored induced subgraph of $G[N_G^3[u]]$. Let D^* be a minimal dominating set of a graph G such that $G[D^*]$ has an isolated vertex. Let also $u \in V(G)\backslash D^*$ and X is a non-empty independent set of $G[D^*]$ such that $X \subseteq D^* \cap N_G(u)$. Consider the set $D' = (D\backslash X) \cup \{u\}$. Denote by **Blue** the set of vertices that

are not dominated by D'. Notice that $\mathbf{Blue} \subseteq N_G(X)\backslash N_G[u]$. Therefore, $\mathbf{Blue} \subseteq N_G^2[u]$. Let $\mathbf{Red} = N_G(\mathbf{Blue})\backslash N_G[X]$. Clearly, $\mathbf{Red} \subseteq N_G^3[u]$. We construct the colored graph $H = G[\mathbf{Red} \cup \mathbf{Blue}]$. Let $\mathcal{A}'$ be an algorithm that enumerates minimal **Red** dominating sets in H. Assume that if $\mathbf{Blue} = \varnothing$, then $\mathcal{A}'$ returns $\varnothing$ as the unique **Red** dominating set. We construct $\mathcal{A}$ as follows.

Step 1. If $\mathcal{A}'$ returns an empty list of sets, then $\mathcal{A}$ returns an empty list as well.

Step 2. For each **Red** dominating set R of H, consider $D'' = D' \cup R$ and construct a minimal dominating set D from D'' by greedy removal.

Lemma 29. *If $\mathcal{A}'$ lists all minimal* **Red** *dominating sets with polynomial delay, then $\mathcal{A}$ generates with polynomial delay a family of minimal dominating sets $\mathcal{D}$ with the property that $\mathcal{D}$ contains all minimal dominating sets D that are children of D^* with respect to flipping u and X.*

Proof. First, we show that $\mathcal{A}$ produces pairwise distinct minimal dominating sets of G. Let R be a **Red** dominating set of H. The set $D' = (D\backslash X) \cup \{u\}$ dominates all vertices of G except the vertices of **Blue**. Since R dominates **Blue**, D'' is a dominating set of G and, therefore, D^* obtained from D'' by the greedy removal is a minimal dominating set. To see that all generated sets are distinct, observe that every vertex of R has its certificate in **Blue**. Therefore, the greedy removal never deletes vertices of R. Since all sets R generated by $\mathcal{A}'$ are pairwise distinct, the claim follows.

Let D be a child of D^* with respect to flipping u and X. Then $D = ((D^* \cup \{u\})\backslash X) \cup Z$. Recall that $u \notin Z$, $X \cap Z = \varnothing$ and the vertices of Z are not adjacent to the vertices of X by the definition of X and Z. Hence, $Z \cap \mathbf{Blue} = \varnothing$. Because D is minimal, each vertex of Z has a certificate. As only the vertices of **Blue** are not dominated by $(D^* \cup \{u\})\backslash X$, each vertex of Z has its certificate in **Blue**. It remains to observe that Z dominates **Blue**, to see that Z is a minimal **Red** dominating set of H. Because $\mathcal{A}'$ generates all minimal **Red** dominating sets, we have that $D \in \mathcal{D}$. □

Now we are ready to prove the main result of the section.

Theorem 30. *For a unit square graph G given with its realization f, all minimal dominating sets of G can be enumerated in incremental polynomial time.*

Proof. It is straightforward to observe that for a vertex u of a unit square graph G, any independent set $X \subseteq N_G(u)$ has at most 4 vertices. Hence, all independent sets $X \subseteq N_G(u)$ for a vertex u can be enumerated in polynomial time. By combining Theorems 4 and 27, and Lemmas 28 and 29, we obtain the claim. □

5 Conclusion

We first presented a linear delay algorithm for enumerating the set of 1-minimal and 1-maximal **Red** (σ, ρ)-dominating sets in colored graphs of bounded LMIM-width, for a fixed pair (σ, ρ) of finite or co-finite subsets of $\mathbb{N}$ (see Theorem 4). We presented also a polynomial time algorithm for counting such sets. As a corollary we generalise the results in [18], and obtain, for instance, new tractable cases for the counting and enumeration of minimal (total) dominating sets including circular-arc graphs, d-trapezoid graphs, complements of d-degenerate graphs, bipartite tolerance graphs, etc. Another consequence is a polynomial time algorithm for counting the minimal transversals in circular-arc hypergraphs, and also enumerate them with linear delay.

In a second step we demonstrated the generality of our first theorem and showed that by combining it with the flipping method introduced in [13] we can enumerate with incremental polynomial delay the set of minimal dominating sets in unit-square graphs, and generally in graphs with locally bounded LMIM-width, provided the independent sets included in the neighborhood of a vertex can be listed in polynomial time.

Our algorithms in the proof of Theorem 4 rely deeply in the tools introduced in [7] to prove that computing a minimum (or maximum) (σ, ρ)-dominating set can be done in polynomial time in many graph classes, including graphs of bounded LMIM-width. We claim that one can extend Theorem 4 to the same graph class considered in [7], and we will give evidences in the next lines. Let us define the graph class introduced in [7].

Definition 31. *A* layout *of a graph $G := (V, E)$ is a pair $(T, \mathcal{L})$ of a tree T and a bijective function $\mathcal{L} : V \to L_T$, with L_T the set of leaves of T. For each edge e of T, the connected components of $T - e$ induce a bipartition $(X_e, V \backslash X_e)$ of L_T, and thus a bipartition $(X^e, V \backslash X^e) = (\mathcal{L}^{-1}(X_e), \mathcal{L}^{-1}(V(T) \backslash X_e))$ of V. A* linear layout *of a finite set V is a layout $(T, \mathcal{L})$ of V such that T is a caterpillar.*

Let $d \in \mathbb{N}$. Let G be a colored graph and let $A \subseteq V_G$. We denote by $nec_R(\equiv_A^d)$ and $nec_B(\equiv_A^d)$ the number of red and blue equivalence classes w.r.t. $\equiv_A^d$. *We let*

$$\text{neigh}_G^d(A) := \max\{nec_R(\equiv_A^d), nec_B(\equiv_A^1), nec_R(\equiv_A^d), nec_B(\equiv_A^1)\}.$$

If $(T, \mathcal{L})$ is a layout of G, then the d-neighborhood-width *of $(T, \mathcal{L})$ is defined as* $\max_{e \in E(T)}\{\text{neigh}_G^d(X^e)\}$.

An n-vertex graph $G := (V, E)$ is said to be of polynomially (linear) bounded neighbourhood *if, for each $d \in \mathbb{N}$, there is a (linear) layout of G of d-neighborhood-width at most $p(n^d)$ for some polynomial p.*

It is not hard to prove that a graph $G := (V, E)$ has a linear layout $(T, \mathcal{L})$ of d-neighborhood-width $\leqslant c$ iff there is a linear ordering $x_1, \ldots, x_n$ of

V such that $neigh_G^d(A_i) \leqslant c$ for each $1 \leqslant i \leqslant n$. Hence, graphs of bounded (L)MIM-width c are of polynomially (linear) bounded neighborhood. The main theorem in [7] is the following[1].

Theorem 32 ([7])**.** *Let (σ, ρ) be a fixed pair of finite or co-finite subsets of $\mathbb{N}$ and let $d := d(\sigma, \rho)$. For a colored graph G given with a layout $(T, \mathcal{L})$ of d-neighborhood-width c one can compute in time $O(c^4 \cdot n)$ a minimum (or maximum)* **Red** *(σ, ρ)-dominating set of G.*

Let (σ, ρ) be a pair of finite or co-finite subsets of $\mathbb{N}$ and let $d := d(\sigma, \rho)$. Let us now recall the important ingredients in our construction. First, if $x_1, x_2, \ldots, x_n$ is a linear ordering of LMIM-width at most k, then $nec(\equiv^d_{A_i}) \leqslant n^{d \cdot k}$ (Lemma 10). Second, if D is a 1-minimal (or 1-maximal) **Red** (σ, ρ)-dominating set, then for each i, there is a 4-tuple $(R, R', C, C') \subseteq LR_i^d \times LR_{\bar{i}}^d \times (A_i)^k \times (\bar{A}_i)^k$ such that $D \cap A_i \equiv^d_{A_i} R$, $D \cap \bar{A}_i \equiv^d_{\bar{A}_i}$, and each vertex in $D \cap A_i$ (resp. $D \cap \bar{A}_i$) has a certificate in $A_i \cup C'$ (resp. $\bar{A}_i \cup C$); and (R, R', C, C') can be chosen in a canonical way.

Now, if $x_1, \ldots, x_n$ is a linear ordering of d-neighborhood-width c, then also $nec(\equiv^d_{A_i}) \leqslant n^{d \cdot k}$ and $nec(\equiv^d_{\bar{A}_i}) \leqslant n^{d \cdot k}$ for some constant k. Further, whenever D is a 1-minimal (or 1-maximal) **Red** (σ, ρ)-dominating set, then for each i, there is a 4-tuple $(R, R', C, C') \subseteq LR_i^d \times LR_{\bar{i}}^d \times 2^{A_i} \times 2^{\bar{A}_i}$ such that $D \cap A_i \equiv^d_{A_i} R$, $D \cap \bar{A}_i \equiv^d_{\bar{A}_i}$, and each vertex in $D \cap A_i$ (resp. $D \cap \bar{A}_i$) has a certificate in $A_i \cup C'$ (resp. $\bar{A}_i \cup C$); C and C' are respectively the certificates in $\bar{A}_i$ and A_i of vertices in $D \cap \bar{A}_i$ and $D \cap A_i$. But, as for vertices in $D \cap A_i$ (resp. in $D \cap \bar{A}_i$) we are only interested in knowing that they have a certificate, and hence a neighbor in C' (resp. in C), we may restrict to choose (C, C') among $LR_i^1 \times LR_{\bar{i}}^1$. Therefore, we can in Sections 3.2 and 3.3 choose the nodes of the DAGs in $LR_i^d \times LR_{\bar{i}}^d \times (LR_i^1 \cap 2^{\mathbf{Blue}}) \times (LR_{\bar{i}}^1 \cap 2^{\mathbf{Blue}})$ and replace $\mathcal{SG}_j(S)$ and $\mathcal{GG}_j(S)$ by respectively $rep^1_{A_j}(S)$ and $rep^1_{\bar{A}_j}(S)$. We can therefore re-state Theorem 4 as follows.

Theorem 33. *Let (σ, ρ) be a pair of finite or co-finite subsets of $\mathbb{N}$ and let c be a positive integer. For a colored graph G given with a linear ordering of $V(G)$ of d-neighborhood-width at most c, one can, after a pre-processing in time $O(c^8 \cdot n^2)$, count in time bounded by $O(c^4 \cdot n)$, and enumerate with linear delay, all 1-minimal (or 1-maximal)* **Red** *(σ, ρ)-dominating sets of G.*

Let us now explain how to extend this theorem to colored graphs given with a layout of d-neighborhood-width at most c. Let us define the notion of *AND-OR DAGS*. To ease the presentation, we will restrict ourselves to 1-minimal **Red** (σ, ρ)-dominating sets.

[1]The original statement dealt with un-colored graphs, however it is not hard to extend it to colored graphs.

Definition 34. *An* AND-OR DAG *is a DAG with one single source, called its* root*, and each internal node is either labeled with* AND *and called* AND-node*, or* OR *and called* OR-node*. Each OR-node corresponds to a branching, and each AND-node has out-degree* 2 *and corresponds, to some extent in our setting, to the way* 1*-minimal* **Red** (σ, ρ)*-dominating sets are constructed from the layout* $(T, \mathcal{L})$*. (In our case the terminals will correspond to the vertices of* G*.) If* H *is an AND-OR DAG and* v *is a node of* H*, we denote by* $H \downarrow v$ *the subgraph of* H *induced by the set of nodes reachable from* v*. The set of* AND-OR trees originating from a node v *of an AND-OR DAG* H *is defined inductively as follows (*$r(F)$ *meaning the root of* F *is a child of* r*).*

(TT) If v *is a terminal, then* v *is an AND-OR tree.*

(TO) If v *is an OR-node and* $v_1, \ldots, v_l$ *are its children, then* $v(F_1), \ldots, v(F_l)$ *are AND-OR trees with* F_i *an AND-OR tree of* $H \downarrow v_i$*.*

(TA) If v *is an AND-node and* v_1, v_2 *are its children, then* $v(F_1, F_2)$ *is an AND-OR tree with* F_i *an AND-OR tree of* $H \downarrow v_i$*.*

The AND-OR trees originating from the source of an AND-OR DAG are called its AND-OR trees.

Theorem 35 ([9])**.** *One can count in linear time and enumerate with linear delay the set of AND-OR trees of any AND-OR-DAG.*

Let $(T, \mathcal{L})$ be a layout of d-neighborhood-width c and let us choose an edge e of T, subdivide it and root the new tree with root the new node r and denote it by T'. Let us define our AND-OR-DAG $AODAG(G)$ as follows.

For each node u of T' and each $(R, R', C, C') \in LR^d_u \times LR^d_{\overline{u}} \times LR^1_u \times LR^1_{\overline{u}}$,

1. (R, R', C, C', u) is an OR-node.
2. If u is an internal node with children v_1 and v_2, then for each $(R_1, R'_1, C_1, C'_1) \in LR^d_{v_1} \times LR^d_{\overline{v_1}} \times LR^1_{v_1} \times LR^1_{\overline{v_1}}$ and $(R_2, R'_2, C_2, C'_2) \in LR^d_{v_2} \times LR^d_{\overline{v_2}} \times LR^1_{v_2} \times LR^1_{\overline{v_2}}$ such that
   (a) $R \equiv^d_{A_u} (R_1 \cup R_2)$, $R'_1 \equiv^d_{\bar{A}_{v_1}} (R' \cup R_2)$ and $R'_2 \equiv^d_{\bar{A}_{v_2}} (R' \cup R_1)$, and
   (b) $C \equiv^1_{A_u} (C_1 \cup C_2)$, $C'_1 \equiv^1_{\bar{A}_{v_1}} (C' \cup C_2)$ and $C'_2 \equiv^1_{\bar{A}_{v_2}} (C' \cup C_1)$,

   then we create an AND-node $(R, R', C, C', R_1, R'_1, C_1, C'_1, R_2, R'_2, C_2, C'_2, u)$ and the arcs from that AND-node to the OR-nodes $(R_1, R'_1, C_1, C'_1, v_1)$ and $(R_2, R'_2, C_2, C'_2, v_2)$, and an arc from (R, R', C, C', u) to that AND-node.
3. If u is a leaf node, then let $x := \mathcal{L}^{-1}(u)$. By definition $R, C \in \{\varnothing, \{x\}\}$. We add the two AND-nodes $(R, R', C, C', \varnothing)$ and $(R, R', C, C', \{x\})$ and

(i) an arc from (R, R', C, C', u) to $(R, R', C, C', \{x\})$ if the following conditions are satisfied
   (a) $R = rep^d_{A_u}(\{x\})$, $|N(x) \cap R'| \in \sigma$ when $x \in$ **Blue**,
   (b) if $x \in$ **Blue** and $|N(x) \cap R'| \in \sigma^-$, then $C = rep^1_{A_u}(\{x\})$ if $|N(x) \cap (\bar{A}_u \cap$ **Red**$)| \neq 0$, otherwise $C = rep^1_{A_u}(\varnothing)$,
   (c) either $|N(x) \cap C'| \neq \varnothing$ or ($x \in$ **Blue** and $|N(x) \cap R'| \in \sigma^*$).

(ii) an arc from (R, R', C, C', u) to $(R, R', C, C', \varnothing)$ if the following conditions are satisfied
   (a) $R = rep^d_{A_u}(\varnothing)$, $|N(x) \cap R'| \in \rho$ when $x \in$ **Blue**,
   (b) If $x \in$ **Blue** and $|N(x) \cap R'| \in \rho^-$, then $C = rep^1_{A_u}(\{x\})$ when $|N(x) \cap (\bar{A}_u \cap$ **Red**$)| \neq 0$, otherwise $C = rep^1_{A_u}(\varnothing)$.

$AODAG(G)$ is easily checkable to be an AND-OR DAG with $(\varnothing, \varnothing, \varnothing, \varnothing, r)$ as root, and with terminals the AND-nodes $(R, R', C, C', \{x\})$ or $(R, R', C, C', \varnothing)$ with x a vertex of G. With the same arguments as in Lemma 13 we can again prove that one can construct $AODAG(G)$ in time $O(c^8 \cdot n^2)$.

Now, the trace of an AND-OR tree will be defined as the union of the fifth components of its terminals. Since, at the construction of the arcs (R, R', C, C', u) to $(R, R', C, C', \{x\})$ we check that x has a certificate (or has a neighbor in C'), then by the same arguments as in the proof of Lemmas 16 and 17, we can prove that the trace of any AND-OR tree is a 1-minimal **Red** (σ, ρ)-dominating set, and with any 1-minimal **Red** (σ, ρ)-dominating set D one can associate an AND-OR tree with trace D.

Why two AND-OR trees cannot give rise to the same trace? By the construction of the arcs we can check that if H_1 and H_2 are two AND-OR trees with same trace, then for each node u of T', if $(R_1, R'_1, C_1, C'_1, u)$ and $(R_2, R'_2, C_2, C'_2, u)$ are nodes respectively of H_1 and H_2 , then $(R_1, R'_1, C_1) = (R_2, R'_2, C_2)$. By using this observation and the condition 2.(b) we can indeed conclude that also $C'_1 = C'_2$. Therefore, two AND-OR trees cannot have the same trace.

It remains to discuss about the delay between two outputs. The AND-OR trees of $AODAG(G)$ have height $O(h)$ with h the height of T', and hence the delay between two outputs is $O(n)$. In order to have a delay depending on the sizes of the traces we need to clean the AND-OR trees H so that if the trace of a subtree of H is the emptyset, we won't enumerate that part of the AND-OR tree. However, such a procedure is explained in [9, Pages 2684-2685], and can be applied to our AND-OR DAG. Therefore, we can state the following.

Theorem 36. *Let (σ, ρ) be a pair of finite or co-finite subsets of $\mathbb{N}$ and let c be a positive integer. For a colored graph G given with a layout $(T, \mathcal{L})$ of G of d-neighborhood-width at most c, one can, after a pre-processing in time $O(c^8 \cdot n^2)$, count in time bounded by $O(c^8 \cdot n)$, and enumerate with linear delay, all 1-minimal (or 1-maximal)* **Red** *(σ, ρ)-dominating sets of G.*

We finish this section with some questions.

1. Given (σ, ρ), can we similarly count and enumerate the set of minimal (or maximal) (σ, ρ)-dominating sets? For which pairs (σ, ρ) do we have that 1-minimal (or 1-maximal) (σ, ρ)-dominating sets coincide with minimal (or maximal) (σ, ρ)-dominating sets?

2. For which pairs (σ, ρ) do we have the existence of pairs (σ', ρ') so that minimal (or maximal) (σ, ρ)-dominating sets correspond to (σ', ρ')-dominating sets?

3. We obtained new tractable cases for the enumeration of minimal dominating sets, however the question of whether we can list the minimal dominating sets in output-polynomial time remains widely unsolved and there are several important graph classes for which nothing is known, e.g., bipartite graphs, circle graphs and weakly chordal graphs.

References

[1] D. Avis, K. Fukuda. Reverse Search for Enumeration. *Discrete Applied Mathematics* 65(1-3), 21–46 (1996).

[2] E.D. Demaine, F.V. Fomin, M. Hajiaghayi, D.M. Thilikos. Bidimensional parameters and local treewidth. *SIAM J. Discrete Math.* 18(3), 501–511 (2004).

[3] E.D. Demaine, M. Hajiaghayi, D.M. Thilikos. The bidimensional theory of bounded-genus graphs. *SIAM J. Discrete Math.* 20(2), 357–371 (2006).

[4] R. Belmonte, M. Vatshelle. Graph classes with structured neighborhoods and algorithmic applications. *Theor. Comput. Sci.* 511, 54–65 (2013).

[5] A. Brandstädt, V.B. Le, J.P. Spinrad. *Graph classes: a survey.* SIAM Monographs on Discrete Mathematics and Applications, SIAM, Philadelphia, PA (1999).

[6] H. Breu. Algorithmic aspects of constrained unit disk graphs. PhD thesis, The University of British Columbia (1996).

[7] B.M. Bui-Xuan, J.A. Telle, M. Vatshelle. Fast dynamic programming for locally checkable vertex subset and vertex partitioning problems. *Theor. Comput. Sci.* 511, 66–76 (2013).

[8] L.S. Chandran, M.C. Francis, N. Sivadasan. On the cubicity of interval graphs. *Graphs and Combinatorics* 25(2), 169–179 (2009).

[9] B. Courcelle. Linear delay enumeration and monadic second-order logic. *Discrete Applied Mathematics* 157, 2675–2700 (2009).

[10] T. Eiter, G. Gottlob. Identifying the minimal transversals of a hypergraph and related problems. *SIAM J. Comput.* 24, 1278–1304 (1995).

[11] T. Eiter, G. Gottlob. Hypergraph transversal computation and related problems in Logic and AI. Proceedings of *JELIA* 2002, *LNCS* 2424, pp. 549–564 (2002).

[12] T. Eiter, G. Gottlob, K. Makino. New results on monotone dualization and generating hypergraph transversals. *SIAM J. Comput.* 32, 514–537 (2003).

[13] P. A. Golovach, P. Heggernes, D. Kratsch, Y. Villanger. An incremental polynomial time algorithm to enumerate all minimal edge dominating sets. *Algorithmica* 72, 836-859 (2015).

[14] P. A. Golovach, P. Heggernes, M. M. Kanté, D. Kratsch, Y. Villanger. Enumerating minimal dominating sets in chordal bipartite graphs. *Discrete Applied Mathematics* 166, 30–35 (2016).

[15] D.S. Johnson, C.H. Papadimitriou, M. Yannakakis. On generating all maximal independent sets. *Inf. Process. Lett.* 27(3), 119–123 (1988).

[16] M. M. Kanté, V. Limouzy, A. Mary, L. Nourine. On the enumeration of minimal dominating sets and related notions. *SIAM J. Discrete Math.* 28, 1916–1929 (2014).

[17] M. M. Kanté, V. Limouzy, A. Mary, L. Nourine. On the Neighbourhood Helly of some Graph Classes and Applications to the Enumeration of Minimal Dominating Sets. Proceedings of *ISAAC 2012*, *LNCS* 7676, pp. 289–298 (2012).

[18] M. M. Kanté, V. Limouzy, A. Mary, and L. Nourine, T. Uno,. On the Enumeration and Counting of Minimal Dominating sets in Interval and Permutation Graphs. Proceedings of *ISAAC 2013*, *LNCS* 8283, pp. 339-349 (2013).

[19] M. M. Kanté, V. Limouzy, A. Mary, L. Nourine, T. Uno. A Polynomial Delay Algorithm for Enumerating Minimal Dominating Sets in Chordal Graphs. Proceedings of *WG 2015*, to appear. arxiv:1407.2036 (2014).

[20] M. M. Kanté, V. Limouzy, A. Mary, L. Nourine, T. Uno. Polynomial Delay Algorithm for Listing Minimal Edge Dominating sets in Graphs. Proceedings of *WADS 2015*, *LNCS* 9214, pp. 446-357 (2015).

[21] L. Khachiyan, E. Boros, K. Borys, K. M. Elbassioni, V. Gurvich. Generating all vertices of a polyhedron is hard. *Discrete & Computational Geometry* 39, 174–190 (2008).

[22] L. Khachiyan, E. Boros, K. M. Elbassioni, V. Gurvich. On enumerating minimal dicuts and strongly connected subgraphs. *Algorithmica* 50, 159–172 (2008).

[23] E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan. Generating all maximal independent sets: NP-hardness and polynomial-time algorithms. *SIAM J. Comput.* 9, 558–565 (1980).

[24] I. Rauf. Polynomially Solvable Cases of Hypergraph Transversal and Related Problems. PhD thesis, Saarland University (2011).

[25] R. E. Tarjan. Enumeration of the elementary circuits of a directed graph. *SIAM J. Comput.* 2, 211–216 (1973).

[26] J.A. Telle, A. Proskurowski. Algorithms for vertex partitioning problems on partial k-trees. *SIAM J. Discrete Math.* 10(4), 529–550 (1997).

[27] M. Vatshelle. New width parameters of graphs. PhD thesis, University of Bergen (2012).