Solving Partial Dominating Set and Related Problems Using Twin-Width

Jakub Balabán
Faculty of Informatics, Masaryk University, Brno, Czechia

Daniel Mock
Dept. of Computer Science, RWTH Aachen University, Aachen, Germany

Peter Rossmanith
Dept. of Computer Science, RWTH Aachen University, Aachen, Germany

Abstract

Partial vertex cover and partial dominating set are two well-investigated optimization problems. While they are W[1]-hard on general graphs, they have been shown to be fixed-parameter tractable on many sparse graph classes, including nowhere-dense classes. In this paper, we demonstrate that these problems are also fixed-parameter tractable with respect to the twin-width of a graph. Indeed, we establish a more general result: every graph property that can be expressed by a logical formula of the form $\phi \equiv \exists x_1 \cdots \exists x_k \sum_{\alpha \in I} \#y\, \psi_\alpha(x_1, \ldots, x_k, y) \geq t$, where ψ_α is a quantifier-free formula for each $\alpha \in I$, t is an arbitrary number, and $\#y$ is a counting quantifier, can be evaluated in time $f(d,k)n$, where n is the number of vertices and d is the width of a contraction sequence that is part of the input. In addition to the aforementioned problems, this includes also connected partial dominating set and independent partial dominating set.

2012 ACM Subject Classification Theory of computation → Parameterized complexity and exact algorithms

Keywords and phrases Partial Dominating Set, Partial Vertex Cover, meta-algorithm, counting logic, twin-width

Funding *Daniel Mock and Peter Rossmanith:* Supported by the Deutsche Forschungsgemeinschaft (DFG, German Science Foundation) – DFG-927/15-2
Jakub Balabán: Brno Ph.D. Talent Scholarship Holder – Funded by the Brno City Municipality

Acknowledgements We would like to thank one of the anonymous reviewers for suggesting a generalization of the logic we considered that also captures the Partial Vertex Cover problem.

1 Introduction

Vertex Cover, Independent Set, and Dominating Set are well-known graph problems that have driven significant advancements in parameterized complexity theory. Despite being NP-complete, these problems motivated the development of new approaches, as the difficulty of finding solutions of size k appears to increase significantly from one problem to the next [15]. On general graphs, Vertex Cover has been found to be fixed-parameter tractable, while Independent Set is W[1]-complete and Dominating Set is W[2]-complete. Consequently, researchers have designed faster algorithms for solving Vertex Cover on general graphs [9, 4, 7, 8, 10, 14, 36, 37, 40], while identifying more and more general graph classes that admit fixed-parameter algorithms for the other two problems. This progression started with planar graphs [1] and led to nowhere-dense graph classes via intermediate results [13, 19, 38, 12]. All three problems can be expressed in first-order logic, and significant advances have been made in meta-algorithm design to solve the first-order model-checking problem. This progression begins with bounded-degree graphs, continues through bounded genus, minor-closed, bounded expansion, nowhere-dense, and finally monadically stable classes [39, 22, 20, 11, 25, 16].

Twin-width is a recently introduced width measure that generalizes clique-width [6]. Classes of bounded twin-width do not contain bounded degree graph classes and are not necessarily monadically stable. They are therefore incomparable to the sparsity hierarchy. Bonnet et al. have developed a parameterized algorithm for first-order model-checking on classes of bounded twin-width [6]. Please note that, unlike tree-width, there is no efficient way known to compute or even approximate twin-width. Moreover, most twin-width based algorithms, including first-order model-checking, require as part of the input not only the graph but also a contraction sequence. The contraction sequence proves that the graph has small twin-width and guides an algorithm. This guidance can be compared to that of dynamic programming on tree decompositions. An optimal tree decomposition can, however, be relatively efficiently computed and even faster approximated. In the world of sparsity algorithms operating in classes of bounded expansion or nowhere-dense classes, the situation is even better: no underlying structure is needed; the algorithm works correctly on every graph but becomes slow when applied outside its scope.

While this represents a significant body of work on Vertex Cover, Independent Set, Dominating Set, and many other first-order expressible problems, the story does not end here. In Vertex Cover, you ask for k nodes that cover all edges, and in Dominating Set, you seek k nodes that dominate all other nodes. Generalizations of these problems are the partial versions: given k and t, are there k nodes that cover t edges or are there k nodes that dominate t nodes? These problems are known as Partial Vertex Cover and Partial Dominating Set; note that in the parameterized setting, k is the parameter. As generalizations, they turn out to be harder than their complete counterparts. For example, while Vertex Cover is fixed-parameter tractable on general graphs, Partial Vertex Cover has been shown to be W[1]-hard [26], and while Dominating Set is fixed-parameter tractable on degenerate graphs, Partial Dominating Set is W[1]-hard [24]. However, it is fixed-parameter tractable with respect to t even on general graphs [29]. Another example of how the partial variant can be harder are c-closed graphs, where non-adjacent vertices share at most c neighbors. Dominating Set is fixed-parameter tractable for bounded c [30], while Partial Dominating Set becomes a hard problem even for $c = 2$ [28].

Another sequence of papers has demonstrated that Partial Dominating Set can be solved in time $f(k)n^{O(1)}$ for larger and larger graph classes. Amini, Fomin, and Saurabh developed an algorithm with a running time of $O(f(k) \cdot t \cdot n^{C_H})$ for the class of H-minor-free graphs, where C_H is a constant dependent only on H. The graph classes with bounded expansion and nowhere-dense graph classes introduced in the sparsity project by Nešetřil and Ossona de Mendez [35] contain all H-minor-free classes. For bounded expansion, there exists a meta-algorithm that solves many problems, including Partial Dominating Set in linear fixed-parameter tractable time, i.e., in $f(k)n$ steps [18]. Additionally, for nowhere-dense classes, there is a more restricted meta-algorithm that is general enough to solve Partial Dominating Set in time $O(f(k)n^{1+\epsilon})$ for every $\epsilon > 0$ [17].

The "simpler" Partial Vertex Cover has not seen as much attention, but has still been investigated as well; see, for example [32, 21, 33].

Our contribution. We demonstrate that both Partial Dominating Set and Partial Vertex Cover can be solved efficiently on graph classes with bounded twin-width. To achieve this, we develop two dynamic programming algorithms that work along a contraction sequence provided as part of the input. Interestingly, both algorithms share similarities in their complexity, while usually Partial Vertex Cover is the much easier problem. The algorithms are based on ideas developed by Bonnet et al. in order to solve the (complete)

Dominating Set problem on bounded twin-width [5].

Having separate algorithms for related problems is not ideal and it would be beneficial to have them emerge as special cases within a more comprehensive framework. Again, the language of logic could facilitate this. We present a meta-algorithm that solves the model-checking problem for formulas of the form $\phi \equiv \exists x_1 \cdots \exists x_k \# y\, \psi(x_1, \ldots, x_k, y) \geq t$ where ψ must be quantifier-free (this logic was considered in [17]). Such a formula is true in a graph $G = (V, E)$ if it contains nodes $v_1, \ldots, v_k$ such that $\psi(v_1, \ldots, v_k, u)$ holds for at least t many nodes $u \in V$. For instance, with $\psi(x_1, \ldots, x_k, y) = \bigvee_{i=1}^{k}(E(x_i, y) \vee x_i = y)$, we can solve Partial Dominating Set. With a quantifier-free ψ, we can express numerous other intriguing problems, including Connected Partial Dominating Set and Independent Partial Dominating Set, as well as more exotic ones like "Are there k nodes that are all part of triangles and dominate at least t of all other nodes?" or "How many vertices do we have to delete in order to get a 2-independent set of size k?" (see [27, 2] for applications). The time needed to decide $G \models \phi$ is $d^{O(k^2 d)}n$, which is linear FPT time. "Linear" means that the input has to be shorter than the classical encoding of the underlying graph as an adjacency list or adjacency matrix. While graphs of bounded twin-width can have a quadratic number of edges, they can still be encoded via their much shorter contraction sequence [6, 23]. Note that our single-purpose algorithm for Partial Dominating Set achieves a better running time of $(kd)^{\mathcal{O}(kd)}n$.

Our meta-algorithm can, in fact, model-check even more general formulas, namely formulas of the form $\phi \equiv \exists x_1 \cdots \exists x_k \sum_{\alpha \in I} \# y\, \psi_\alpha(x_1, \ldots, x_k, y) \geq t$, where ψ_α is a quantifier-free formula for each $\alpha \in I$. Such a formula ϕ is true if the *sum* of the numbers of vertices y satisfying the formulas ψ_α is at least t. Using such a formula, we can also express the Partial Vertex Cover problem, see Example 17.

Finally, let us remark that if we take a power of a graph, then its twin-width remains small [6]. This simple observation allows us to tackle the "distance-r" variants of Partial Dominating Set as well by using transductions.

2 Preliminaries

For integers i and j, we let $[i, j] := \{n \in \mathbb{N} \colon i \leq n \leq j\}$ and $[i] := [1, i]$. We study finite, simple and undirected graphs with non-empty vertex set.

Twin-width A *trigraph* G is a graph whose edge set is partitioned into a set of *black* and *red* edges. The set of red edges is denoted $R(G)$, and the set of black edges $B(G)$; standard graphs are viewed as trigraphs with no red edges. The *red graph* G^R of G is the graph $(V(G), R(G))$ and the *black graph* G^B of G is the graph $(V(G), B(G))$. Any graph-theoretic notion with a color adjective (black or red) has its standard meaning applied in the corresponding graph. For example, the *red degree* of $u \in V(G)$ in G is its degree in G^R.

Given a trigraph G, a *contraction* of two distinct vertices $u, v \in V(G)$ is the operation which produces a new trigraph by (1) removing u, v and adding a new vertex w, (2) adding a black edge wx for each $x \in V(G)$ such that $xu, xv \in B(G)$, and (3) adding a red edge wy for each $y \in V(G)$ such that $yu \in R(G)$, or $yv \in R(G)$, or y contains only a single black edge to either v or u. A sequence $C = (G = G_1, \ldots, G_n)$ is a *contraction sequence of* G if it is a sequence of trigraphs such that $|V(G_n)| = 1$ and for all $i \in [n-1]$, G_{i+1} is obtained from G_i by contracting two vertices. The *width* of a contraction sequence C is the maximum red degree over all vertices in all trigraphs in C. The *twin-width* of G is the minimum width of any contraction sequence of G. An example of a contraction sequence is provided in Figure 1.

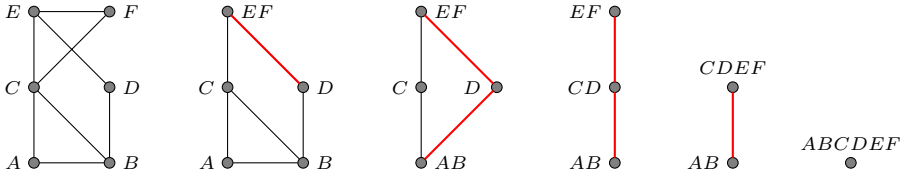

Figure 1 A contraction sequence of width 2 for the leftmost graph, consisting of 6 trigraphs.

If a contraction sequence $C = (G_1, \dots, G_n)$ of G is clear from context, then for each $i \in [n]$ and $u \in V(G_i)$, we call the set of vertices of G contracted to u the *bag* of u and we denote it $\beta(u)$. Formally, $\beta(u) = \{u\}$ if $u \in V(G)$, and $\beta(u) = \beta(v) \cup \beta(w)$ if there is $i \in [2, n]$ such that $u \in V(G_i)$, $v, w \in V(G_{i-1})$, and u is created by contracting v and w. Note that if a vertex appears in multiple trigraphs in C, then its bag is the same in all of them. If $T \subseteq V(G_i)$ for some $i \in [n]$, then we use $\beta(T)$ to denote $\bigcup_{u \in T} \beta(T)$.

Basic profiles Now we define *basic profiles*, inspired by the k-profiles in the Dominating Set algorithm from [5]. Intuitively, a profile describes a "type" of a subsolution we compute during our dynamic algorithms. For each profile, we will compute a subsolution that is, in some sense, best for the profile. In the end, we can read the answer to our problem from the profiles associated with the last trigraph G_n in the contraction sequence.

In the following sections, we will expand the basic profiles with extra information in two different ways: to *extended profiles*, which we will use to solve Partial Dominating Set and Partial Vertex Cover in Section 3, and to *virtual profiles*, which we will use for the model-checking in Section 4. Note that the k-profiles in [5] can also be viewed as extensions of our basic profiles. After defining the basic profiles, we will prove some of their properties, which we will later generalize to the extended and virtual profiles.

Let us fix a graph G, a contraction sequence $(G = G_1, \dots, G_n)$ of width d, and an integer $k \in \mathbb{N}$.

▶ **Definition 1.** If $i \in [n]$, then a *basic i-profile* is a pair (T, D) such that $T \subseteq V(G_i)$, $|T| \leq k(d+1)$, T is connected in G_i^R, $D \subseteq T$, and $|D| \leq k$.

For example, in the Partial Vertex Cover algorithm, k will be the number of covering vertices, D will be the set such that the covering vertices are in $\beta(D)$, and T is the "context" in which the cover is considered, i.e., we are interested in which edges of $G[\beta(T)]$ are covered (this intuition will be formalized in Definition 5).

Now we define how a basic i-profile can be decomposed into several basic $(i-1)$-profiles, again inspired by [5]. Later, we will analogously define decompositions for extended and virtual profiles.

Let $i \in [2, n]$, let $v \in V(G_i)$ be the new vertex in G_i, let $u_1, u_2 \in V(G_{i-1})$ be the two vertices contracted into v, let $\pi = (T, D)$ be a basic i-profile such that $v \in T$, and let $T' = (T \setminus v) \cup \{u_1, u_2\}$. We say that a set $D' \subseteq T'$ is *compatible* with π if $D \setminus \{v\} = D' \setminus \{u_1, u_2\}$, $|D'| \leq k$, and $v \in D$ if and only if $u_1 \in D'$ or $u_2 \in D'$. Note that the condition $|D'| \leq k$ is necessary only to prohibit the case $|D| = k$ and $u_1, u_2 \in D'$.

We say that a set $\{(T_1, D_1), \dots, (T_m, D_m)\}$ of basic $(i-1)$-profiles is a *D'-decomposition* of π, see Figure 2, if:

1. The sets $T_1, \dots, T_m$ are pairwise disjoint.
2. $T' = \bigcup_{j \in [m]} T_j$, $D' = \bigcup_{j \in [m]} D_j$, and D' is compatible with π.
3. If $x \in T_j$ and $y \in D_\ell$ for $j \neq \ell$, then $xy \notin R(G_{i-1})$.

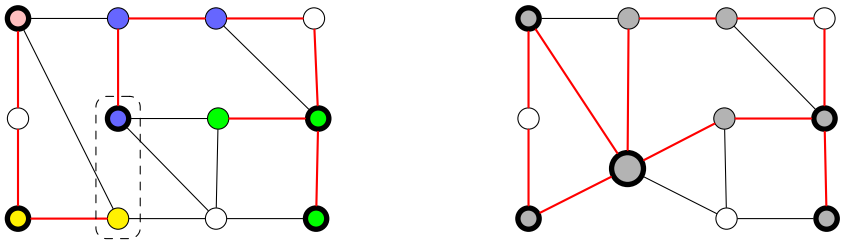

Figure 2 Right: A basic i-profile $\pi = (T, D)$. Vertices of T are colored in gray, and vertices of D have thick boundary. The new vertex of G_i is represented by the bigger disc. **Left:** A D'-decomposition $\{(T_1, D_1), \ldots, (T_4, D_4)\}$ of π. The sets $T_1, \ldots, T_4$ are represented by the colors blue, green, yellow, and pink. Vertices of D' have thick boundary, and the two vertices to be contracted are inside a dashed rectangle.

Let us give some intuition behind condition 3. If $y \in D_\ell$, then there is a dominating (resp. covering) vertex $y' \in \beta(y)$. If $xy \in R(G_{i-1})$, then we would not know which vertices of $\beta(x)$ are dominated by y' (resp. how many edges in $\{x'y' \colon x' \in \beta(x)\}$ are covered by y'). Hence, the connections between profiles must be homogeneous (either a black edge or a non-edge).

Now let us prove that a small decomposition always exists (and can be found efficiently).

▶ **Lemma 2.** *If $i \in [2, n]$, $V(G_i) \setminus V(G_{i-1}) = \{v\}$, $V(G_{i-1}) \setminus V(G_i) = \{u_1, u_2\}$, $\pi = (T, D)$ is a basic i-profile such that $v \in T$, $T' = (T \setminus v) \cup \{u_1, u_2\}$, and $D' \subseteq T'$ is compatible with π, then there is a D'-decomposition $\mathcal{D}$ of π containing at most $d + 2$ basic $(i-1)$-profiles. Moreover, $\mathcal{D}$ can be found in time $\mathcal{O}(kd^2)$.*

Proof. Let $\{T_1, \ldots, T_m\}$ be the connected components of T' in G_{i-1}^R and let $D_j = T_j \cap D'$ for each $j \in [m]$. Observe that (T_j, D_j) is a basic $(i-1)$-profile if and only if $|T_j| \leq k(d+1)$. Hence, if $|T_j| \leq k(d+1)$ for each $j \in [m]$, then $\{(T_1, D_1), \ldots, (T_m, D_m)\}$ is a D'-decomposition of π; condition 3 is satisfied because there are no red edges between distinct T_j and T_ℓ. Let $j \in [m]$ and observe that since T is connected in G_i^R, there is a vertex $u \in T_j$ such that either $uv \in R(G_i)$ or $u \in \{u_1, u_2\}$. Since v has red degree at most d in G_i, we have $m \leq d + 2$ as required.

Now suppose that $|T_j| > k(d+1)$ for some $j \in [m]$. Since $|T| \leq k(d+1)$ and $T_j \subseteq T'$, we deduce that $|T_j| = k(d+1) + 1$, $m = j = 1$, and $(T_j, D_j) = (T', D')$. Since each vertex in D' has red degree at most d in G_{i-1} and $|D'| \leq k$, there is a vertex $w \in T' \setminus D'$ such that $wx \notin R(G_{i-1})$ for each $x \in D'$. Let $\{T'_1, \ldots, T'_p\}$ be the connected components of $T' \setminus \{w\}$ in G_{i-1}^R and let $D'_j = T'_j \cap D'$ for each $j \in [p]$. Now observe that $\{(T'_1, D'_1), \ldots, (T'_p, D'_p)\} \cup \{(\{w\}, \emptyset)\}$ is a D'-decomposition of π. Since T'_j contains a vertex adjacent to w in G_{i-1}^R for each $j \in [p]$, we have $p + 1 \leq d + 2$ as required.

Observe that the proof naturally translates into an algorithm finding the desired decomposition and that all the operations can be performed in time linear in $|T'| + |R(G_{i-1}[T'])|$. Since $|T'| \leq k(d+1) + 1$ and each vertex of T' has red degree at most d in G_{i-1}, there are $\mathcal{O}(kd^2)$ red edges in $G_{i-1}[T']$, and so the stated running time is achieved. ◀

Now we prove that the number of basic i-profiles containing any fixed vertex of G_i is small.

▶ **Lemma 3.** *If $i \in [n]$ and $v \in V(G_i)$, then the number of basic i-profiles (T, D) such that $v \in T$ is in $d^{\mathcal{O}(kd)}$. Moreover, such profiles can be enumerated in time $d^{\mathcal{O}(kd)}$.*

Proof. Let (T, D) be a basic i-profile such that $v \in T$. By Corollary 3.2. from [5], there are at most $d^{2\ell-2} + 1$ red-connected subsets of $V(G_i)$ of size at most ℓ containing v. Since

$|T| \leq k(d+1)$, there are $d^{2kd+2k-2}+1$ possible choices of T. Since $D \subseteq T$, there are $2^{k(d+1)}$ possible choices of D. Hence, there are $d^{\mathcal{O}(kd)}$ basic i-profiles containing v. The claim about enumeration follows from Corollary 3.2. as well. ◀

3 Partial Dominating Set and Vertex Cover

In this section, we show that the following parameterized problems are fixed-parameter tractable. When a graph G is clear from context, we denote the closed neighborhood of $S \subseteq V(G)$ in G by $N[S]$, i.e., $N[S] = S \cup \{u \in V(G) \colon u \text{ has a neighbor in } S\}$.

Partial Dominating Set using twin-width
Input: A graph G, a contraction sequence for G of width d, integers k and t
Question: Is there a set $S \subseteq V(G)$ of size k such that $|N[S]| \geq t$?
Parameter: $k+d$

Partial Vertex Cover using twin-width
Input: A graph G, a contraction sequence for G of width d, integers k and t
Question: Is there a set $S \subseteq V(G)$ of size k such that at least t edges have an endpoint in S?
Parameter: $k+d$

Henceforth, we refer to these problems simply as Partial Dominating Set and Partial Vertex Cover.

3.1 Extended Profiles

For both of these problems, we will need a notion of a profile, obtained by extending basic profiles introduced in Section 2. Let us fix a graph G, a contraction sequence $(G = G_1, \ldots, G_n)$ of width d, and an integer k.

▶ **Definition 4.** Let $i \in [n]$ and let $\pi = (T, D)$ be basic i-profile. An *extended i-profile* is a tuple $\pi' = (T, D, M, f)$ such that $M \subseteq T$ and $f \colon T \to [0, k]$ is a function such that $\sum_{u \in T} f(u) \leq k$ and for each $u \in T$, $f(u) \leq |\beta(u)|$, and $D = \{u \in T \colon f(u) \neq 0\}$.

Informally, the function f specifies the distribution of the dominating (resp. covering) vertices across D, and the set M specifies which bags we want to dominate vertices in. Note that for Partial Vertex Cover, M will not be used, i.e., $M = \emptyset$ will always hold (this allows us to use the same definition of a profile for both problems). The following definition formalizes this idea.

▶ **Definition 5.** Let $i \in [n]$ and let $\pi = (T, D, M, f)$ be an extended i-profile. A set $S \subseteq \beta(T)$ is a *solution* of π if for every $u \in T$, $f(u) = |S \cap \beta(u)|$. We define the *dominating-set-value* of S in π as $|N[S] \cap \beta(M)|$ and the *vertex-cover-value* of S in π as $|\{uv \in E(G) \colon u \in S, v \in \beta(T)\}|$.

Note that each set $S \subseteq V(G)$ can be a solution of more than one extended i-profile for each $i \in [n]$. Indeed, S determines only D and f, whereas there are more choices of T and M (T can be any small red-connected set containing D). However, it will always be clear from context which profile a solution belongs to, and so we will often simply say, e.g., *the dominating-set-value* of S (omitting "in π").

Let us give a quick informal overview of how the profiles will be used to solve Partial Dominating Set. During the algorithm, we proceed through each time i ranging from 1 to n and compute for each extended i-profile π its optimal value; that is, the maximum value t such that a solution of π dominates t vertices from $\beta(M)$. To compute the optimal value, we need to decompose the profile: each decomposition is a small set of extended $(i-1)$-profiles,

see Section 3.2 for a formal definition. Crucially, each solution of π is described by some decomposition, and the best solution described by a decomposition $\mathcal{D} = \{\pi_1, \ldots, \pi_m\}$ can be computed from the optimal solutions of the profiles in $\mathcal{D}$ (which were computed in the previous iteration). Hence, the optimal solution of π can be computed by trying all possible decompositions. More precisely, we do not try *all* decompositions but only one decomposition of each "type". Finally, to determine how many vertices can be dominated by a set of size k in G, it suffices to examine the optimal value of one particular n-profile π (note that G_n consists of a single vertex, and so it is easy to define π explicitly). We get this value at the end of the run of Algorithm 1.

Let us now show that the number of extended i-profiles containing any fixed vertex of G_i is bounded.

▶ **Lemma 6.** *If $i \in [n]$ and $v \in V(G_i)$, then the number of extended i-profiles (T, D, M, f) such that $v \in T$ is in $2^{\mathcal{O}(kd \log(kd))}$. Moreover, such profiles can be enumerated in time $2^{\mathcal{O}(kd \log(kd))}$.*

Proof. By Lemma 3, there are $d^{\mathcal{O}(kd)}$ basic i-profiles (T, D). Recall that $|T| \leq k(d+1)$. Since $M \subseteq T$, there are $2^{k(d+1)}$ possible choices of M. Since f is a function of type $T \to [0, k]$, there are at most $(k+1)^{k(d+1)}$ possible choices of f. Hence, there are $d^{\mathcal{O}(kd)} \cdot 2^{\mathcal{O}(kd)} \cdot k^{\mathcal{O}(kd)} = 2^{\mathcal{O}(kd \log(kd))}$ extended i-profiles containing v, and the claim about enumeration follows from Lemma 3. ◀

3.2 Extended Decompositions

Now we generalize the D'-decompositions defined in Section 2 to extended profiles.

Let $i \in [2, n]$, let $v \in V(G_i)$ be the new vertex in G_i, let $u_1, u_2 \in V(G_{i-1})$ be the two vertices contracted into v, let $\pi_0 = (T, D)$ be a basic i-profile such that $v \in T$, let $T' = T \setminus \{v\} \cup \{u_1, u_2\}$, and let $\pi = (T, D, M, f)$ be an extended i-profile. Let $\hat{M} = M \setminus \{v\} \cup \{u_1, u_2\}$ if $v \in M$, and $\hat{M} = M$ otherwise. We say that a function $f' \colon T' \to [0, k]$ is *compatible* with π if $f(v) = f'(u_1) + f'(u_2)$, $f'(u_1) \leq |\beta(u_1)|$, $f'(u_2) \leq |\beta(u_2)|$, and for each $u \in T \setminus \{v\}$, $f(u) = f'(u)$. Note that there is at least one function compatible with π because $f(v) \leq |\beta(v)|$.

Let us fix a function f' compatible with π, let $D' = \{u \in T' \colon f'(u) \neq 0\}$, and let M' be the set defined as follows.

$$M' = \{u \in \hat{M} \colon u \text{ has no black neighbor in } D' \text{ in } G_{i-1}\}. \tag{1}$$

We say that a set of extended $(i-1)$-profiles $\mathcal{D} = \{(T_j, D_j, M_j, f_j) \colon j \in [m]\}$ is an *f'-decomposition* of π if $f' = \bigcup_{j \in [m]} f_j$, $D' = \bigcup_{j \in [m]} D_j$, $M' = \bigcup_{j \in [m]} M_j$, and $\{(T_1, D_1), \ldots, (T_m, D_m)\}$ is a D'-decomposition of π_0.

▶ Remark 7. Equation 1 allows us to avoid double-counting in the algorithm for Partial Dominating Set. Indeed, if $u \in \hat{M}$ has a black neighbor w in D', then all vertices in $\beta(u)$ are dominated by some vertex in $\beta(w)$. Hence, if we added u to some M_j, then vertices in $\beta(u)$ would be counted as dominated twice.

Now we prove that a small f'-decomposition exists for each f' compatible with π.

▶ **Lemma 8.** *If $i \in [2, n]$, $V(G_i) \setminus V(G_{i-1}) = \{v\}$, $V(G_{i-1}) \setminus V(G_i) = \{u_1, u_2\}$, $\pi = (T, D, M, f)$ is an extended i-profile such that $v \in T$, $T' = T \setminus \{v\} \cup \{u_1, u_2\}$, and $f' \colon T' \to [0, k]$ is a function compatible with π, then there is an f'-decomposition $\mathcal{D}$ of π with cardinality at most $d + 2$. Moreover, $\mathcal{D}$ can be found in time $\mathcal{O}(k^2 d^2)$.*

Proof. Let $D' = \{u \in T' \colon f'(u) \neq 0\}$, let $\hat{M} = M \setminus \{v\} \cup \{u_1, u_2\}$ if $v \in M$, and $\hat{M} = M$ otherwise, and let $M' \subseteq T'$ be the set satisfying Equation 1; note that M' is determined by D', which in turn depends on f'. By Lemma 2, there is a D'-decomposition $\mathcal{D}_0 = \{(T_1, D_1), \ldots, (T_m, D_m)\}$ of the basic i-profile (T, D) such that $m \leq d + 2$. For each $j \in [m]$, let $M_j = M' \cap T_j$ and let f_j be the restriction of f' to T_j. It can be easily observed that $\{(T_j, D_j, M_j, f_j) \colon j \in [m]\}$ is the desired f'-decomposition of π.

By Lemma 2, $\mathcal{D}_0$ can be found in time $\mathcal{O}(kd^2)$. Since $|\hat{M}| \leq |T'| \leq k(d+1)+1$ and $|D'| \leq k$, there are $\mathcal{O}(k^2 d)$ possible black edges one has to inspect when computing M'. Once M' is computed, deriving M_j (and f_j) for each $j \in [m]$ is trivial, which implies the stated running time. ◀

The following lemma shows that a solution of a profile π can be constructed by "merging" the solutions of profiles in a decomposition of π.

▶ **Lemma 9.** *If $i \in [2, n]$, π is an extended i-profile, $\{\pi_j \colon j \in [m]\}$ is an f'-decomposition of π, and S_j is a solution of π_j for each $j \in [m]$, then $S := \bigcup_{j \in [m]} S_j$ is a solution of π.*

Proof. Let $\pi = (T, D, M, f)$ and for each $j \in [m]$, let $\pi_j = (T_j, D_j, M_j, f_j)$. By Definition 5, $S_j \subseteq \beta(T_j)$ for each $j \in [m]$. Since $T_1, \ldots, T_m$ are pairwise disjoint, $S_1, \ldots, S_m$ are pairwise disjoint as well. Observe that $S \subseteq \bigcup_{j \in [m]} \beta(T_j) = \beta(T)$ as required.

Now we need to show that $f(u) = |S \cap \beta(u)|$ for each $u \in T$. Let $v \in V(G_i)$ be the new vertex in G_i and let $u_1, u_2 \in V(G_{i-1})$ be the two vertices contracted into v. First, let $u \in T \setminus \{v\}$. Observe that there is a unique $j \in [m]$ such that $u \in T_j$ and $f_j(u) = f'(u) = f(u)$ because f' is compatible with π. Since S_j is a solution of π_j, we have $f(u) = |S_j \cap \beta(u)| = |S \cap \beta(u)|$ as required. Second, let $j, \ell \in [m]$ be such that $u_1 \in T_j$ and $u_2 \in T_\ell$. Since f' is compatible with π, we have $f(v) = f'(u_1) + f'(u_2)$. Since S_j (resp. S_ℓ) is a solution of π_j (resp. π_ℓ), we obtain $f(v) = |S_j \cap \beta(u_1)| + |S_\ell \cap \beta(u_2)| = |S \cap (\beta(u_1) \cup \beta(u_2))| = |S \cap \beta(v)|$ as required. ◀

3.3 Partial Dominating Set

Let $(G, C = (G_1, \ldots, G_n), k, t)$ be an instance of the Partial Dominating Set problem; recall that C is a contraction sequence of width d. In this section, we show that Algorithm 1 solves the problem in time $(dk)^{\mathcal{O}(dk)} n$, see Theorem 13. For an outline of the algorithm, see Section 3.1.

We will use the notion of extended profiles, see Definition 4, and the notion of solutions, see Definition 5. Instead of the dominating-set-value of a solution, we will say simply the *value*. We say that a solution S is an *optimal* solution of an extended profile π if no other solution of π has higher value than S.

Before presenting the algorithm, let us make two key observations.

▶ **Lemma 10.** *Let $i \in [2, n]$, π be an extended i-profile, $\{(T_j, D_j, M_j, f_j) \colon j \in [m]\}$ be an f'-decomposition of π, S be a solution of π, and $S_j = S \cap \beta(T_j)$ for each $j \in [m]$. For each $j \in [m]$:*

a) *The value of S_j equals $|N[S] \cap \beta(M_j)|$.*

b) *If $u \in T_j$ has a black neighbor in D_ℓ for some $\ell \in [m]$, then $\beta(u) \subseteq N[S]$.*

Proof. For a), let us fix $j \in [m]$. It suffices to prove that $N[S_j] \cap \beta(M_j) = N[S] \cap \beta(M_j)$. Suppose for contradiction that there are vertices $u_0 \in \beta(M_j)$ and $v_0 \in S \setminus S_j$ such that $u_0 v_0 \in E(G)$. Let $u, v \in V(G_{i-1})$ be such that $u_0 \in \beta(u)$ and $v_0 \in \beta(v)$, and observe that $v \in D_\ell$ for some $\ell \in [m]$, $\ell \neq j$. Since $u \in M_j$, uv is not a black edge in G_{i-1}, see

Equation (1). However, uv is not a red edge in G_{i-1} either, by condition 3 in the definition of a basic decomposition in Section 2. This is a contradiction with $u_0v_0 \in E(G)$.

For b), let us fix $j \in [m]$ and $u \in T_j$ that has a black neighbor $v \in D_\ell$ for some $\ell \in [m]$. By Definition 4, $f'(v) \neq 0$ and there is a vertex $w \in \beta(v) \cap S$. Hence, $\beta(u) \subseteq N(w) \subseteq N[S]$ as required. ◀

Algorithm 1 Algorithm for Partial Dominating Set

Input: A graph G, contraction sequence $(G_1 = G, \ldots, G_n)$ of width d, and $k, t \in \mathbb{N}$.

1 **for each** *extended* 1*-profile* $\pi = (T, D, M, f)$ **do** $\sigma_1(\pi) := (D, |D \cap M|)$
2 **for** i **from** 2 **to** n **do**
3 $v :=$ the new vertex in G_i
4 $u_1, u_2 :=$ the two vertices of G_{i-1} contracted into v
5 **for each** *extended* i*-profile* $\pi = (T, D, M, f)$ **do**
6 **if** $v \notin T$ **then** $\sigma_i(\pi) := \sigma_{i-1}(\pi)$; **continue** with next π
7 $\sigma_i(\pi) := (\mathsf{null}, -1)$
8 $T' := (T \setminus v) \cup \{u_1, u_2\}$
9 **if** $v \in M$ **then** $\hat{M} := M \setminus \{v\} \cup \{u_1, u_2\}$ **else** $\hat{M} := M$
10 **for each** *function* $f' \colon T' \to [0, k]$ *compatible with* π **do**
11 Let $\{\pi_j \colon j \in [m]\}$ be an f'-decomposition of π s.t. $m \leq d + 2$; it exists by Lemma 8.
12 **for each** $j \in [m]$ **do**
13 $(T_j, D_j, M_j, f_j) := \pi_j$; $(S_j, \nu_j) := \sigma_{i-1}(\pi_j)$
14 $C_j = \{u \in \hat{M} \cap T_j \colon u$ has a black neighbor in D_ℓ for some $\ell \in [m]\}$
15 $\mathsf{value}_j := \nu_j + \sum_{u \in C_j} |\beta(u)|$
16 $\nu := \sum_{j \in [m]} \mathsf{value}_j$; $S := \bigcup_{j \in [m]} S_j$
17 **if** $\nu \geq \nu_0$, *where* $(S_0, \nu_0) = \sigma_i(\pi)$ **then** $\sigma_i(\pi) := (S, \nu)$
18 $(S, \nu) := \sigma_n(\{u\}, \{u\}, \{u\}, \{(u, k)\})$, where $\{u\} := V(G_n)$
19 **return** $\nu \geq t$

Now we show that in each iteration of the loop on lines 10-17 in Algorithm 1, a solution of the currently processed profile π and its value are computed. Informally, for each profile π_j in the decomposition of π, we avoid double-counting dominated vertices in $\beta(\hat{M} \cap T_j)$ by partitioning $\hat{M} \cap T_j$ into two subsets, M_j and C_j: $\beta(M_j)$ contains vertices dominated "from inside", i.e., via a red edge considered in one of the previous iterations, and $\beta(C_j)$ contains those dominated "from outside", i.e., via a black edge to D'.

▶ **Lemma 11.** *Let $i \in [2, n]$, π be an extended i-profile, f' be a function compatible with π, and ν and S be the values computed by Algorithm 1 on line 16 in the iteration processing f'. If for each extended $(i-1)$-profile π', S' is a solution of π' of value ν', where (S', ν') is the pair computed in $\sigma_{i-1}(\pi')$, then S is a solution of π of value ν.*

Proof. Let $\pi = (T, D, M, f)$ and let $\{\pi_j \colon j \in [m]\}$ be the f'-decomposition of π found on line 11 in the iteration processing f'. For each $j \in [m]$, let $T_j, D_j, M_j, f_j, S_j, \nu_j$, and C_j be the values computed on lines 13 and 14. By Lemma 9, S is a solution of π. To simplify notation, let $\delta(A) := |N[S] \cap \beta(A)|$ for any $A \subseteq V(G_i)$ or $A \subseteq V(G_{i-1})$. Now it suffices to show that $\nu = \delta(M)$.

Let $\hat{M}$ be defined as in the algorithm, see line 9, and observe that $\delta(M) = \delta(\hat{M})$. Since $T_1, \ldots, T_m$ are pairwise disjoint, it suffices to show that for each $j \in [m]$, value_j computed on line 15 equals $\delta(\hat{M} \cap T_j)$. Let us fix $j \in [m]$, and observe that $\hat{M} \cap T_j = M_j \cup C_j$ and $M_j \cap C_j = \emptyset$, see Equation (1) and line 14. By Lemma 10a, we have $\nu_j = \delta(M_j)$ because ν_j is the value of S_j. By Lemma 10b, $\beta(u) \subseteq N[S]$ for each $u \in C_j$, which implies $\delta(C_j) = |\beta(C_j)|$. Hence, $\mathsf{value}_j = \delta(M_j) + \delta(C_j) = \delta(\hat{M} \cap T_j)$, and $\nu = \delta(M)$ as required. ◀

Now we show that we are able to find an *optimal* solution for each profile. The idea is to describe an optimal solution with a function f' compatible with the considered profile, and then to show that the solution found in the iteration processing f' is optimal as well.

▶ **Lemma 12.** *For each $i \in [n]$ and each extended i-profile π, if the final value of $\sigma_i(\pi)$ computed by Algorithm 1 is (S, ν), then S is an optimal solution of π and ν is the value of S.*

Proof. Let $i \in [n]$ and $\pi = (T, D, M, f)$ be an extended i-profile. We proceed by induction on i. First, suppose that $i = 1$. Since there are no red edges in $G_1 = G$ and T is connected in G^R, we know that $T = \{u\}$ for some $u \in V(G)$. Notice that $D, M \in \{\emptyset, \{u\}\}$ and $f \in \{\{(u,0)\}, \{(u,1)\}\}$. In particular, π has only one solution, namely D, and the value of D is $|D \cap M|$ as required, see line 1. Now suppose that $i > 1$. Let $v \in V(G_i)$ be the new vertex in G_i and let $u_1, u_2 \in V(G_{i-1})$ be the two vertices contracted into v. If $v \notin T$, then π is also an $(i-1)$-profile. By induction hypothesis, $\sigma_{i-1}(\pi)$ is an optimal solution of π, which ensures that $\sigma_i(\pi)$ is also an optimal solution of π, see line 6. Hence, let us assume that $v \in T$ and let $T' = (T \setminus v) \cup \{u_1, u_2\}$. By induction hypothesis and Lemma 11, π has a solution since there is a function compatible with π.

Let S^* be an optimal solution of π, let ν^* be the value of S^*, and let $f' \colon T' \to [0, k]$ be defined as $f'(u) = |S^* \cap \beta(u)|$. Observe that by Definition 5, $f'(u) = f(u)$ for each $u \in T \setminus \{v\}$, and $f(v) = |S^* \cap \beta(v)| = |S^* \cap \beta(u_1)| + |S^* \cap \beta(u_2)| = f'(u_1) + f'(u_2)$, i.e., f' is compatible with π. Let us consider the iteration of the loop on lines 5-17 processing π and the iteration of the loop on lines 10-17 processing f'. Let $\{\pi_j \colon j \in [m]\}$ be the f'-decomposition of π found on line 11. For each $j \in [m]$, let $T_j, D_j, M_j, f_j, S_j, \nu_j$, and C_j be as in the algorithm. Let ν and S be the values computed on line 16. By Lemma 11, S is a solution of π with value ν. Similarly to the proof of Lemma 11, we denote $\delta(A) = |N[S] \cap \beta(A)|$ and $\delta^*(A) = |N[S^*] \cap \beta(A)|$ for any $A \subseteq V(G_i)$ or $A \subseteq V(G_{i-1})$.

Suppose for contradiction that $\nu < \nu^*$, i.e., $\delta(M) < \delta^*(M)$, see Definition 5. Let $\hat{M}$ be as in the algorithm, see line 9, and observe that $\delta(M) = \delta(\hat{M})$ and $\delta^*(M) = \delta^*(\hat{M})$. Hence, there must be $j \in [m]$ such that $\delta(\hat{M} \cap T_j) < \delta^*(\hat{M} \cap T_j)$; let us fix such a j. By Lemma 10b, if $u \in C_j$, then $\beta(u) \subseteq N[S] \cap N[S^*]$. Hence, $\delta(C_j) = |\beta(C_j)| = \delta^*(C_j)$, which implies $\delta(M_j) < \delta^*(M_j)$ because $\hat{M} \cap T_j = C_j \cup M_j$ and $M_j \cap C_j = \emptyset$. Let $S_j^* = S^* \cap \beta(T_j)$ and observe that S_j^* is a solution of π_j. By Lemma 10a, $\delta(M_j)$ is the value of S_j and $\delta^*(M_j)$ is the value of S_j^*. By induction hypothesis, S_j is an optimal solution of π_j, which is a contradiction with $\delta(M_j) < \delta^*(M_j)$. Hence, $\nu = \nu^*$, and S is an optimal solution of π, too.

After the iteration of the loop on lines 10-17 processing f', $\sigma_i(\pi) = (S, \nu)$ because S is an optimal solution of π. Observe that the second component of $\sigma_i(\pi)$ cannot change in any of the following iterations because ν is always the value of some solution of π by Lemma 11. Hence, after all functions compatible with π have been processed, $\sigma_i(\pi)$ is indeed an optimal solution of π and its value, as desired. ◀

Using Lemma 12, it is easy to show the correctness of Algorithm 1.

▶ **Theorem 13.** *Algorithm 1 decides the Partial Dominating Set problem in time $2^{\mathcal{O}(kd \log(kd))} \cdot n$.*

Proof. Let $\iota = (G, C = (G = G_1, \ldots, G_n), k, t)$ be the input instance, where C is contraction sequence of width d. Let $V(G_n) = \{u\}$ and let $\pi = (\{u\}, \{u\}, \{u\}, \{(u, k)\})$. Note that π is an extended n-profile unless $n < k$, in which case, ι is trivially a NO instance. By Lemma 12, S is an optimal solution of π with value ν, where (S, ν) is the pair computed by Algorithm 1 on line 18. By Definition 5, $|S| = k$ and $\nu = |N[S] \cap \beta(\{u\})| = |N[S]|$. Hence, ι is a YES instance if and only if $\nu \geq t$, as required.

Now need to argue that Algorithm 1 achieves the desired running time. As an optimization, we maintain a single function σ (instead of σ_i for each $i \in [n]$). Since there are $\mathcal{O}(n)$ extended 1-profiles, line 1 can be performed in time $\mathcal{O}(n)$. There are $n - 1$ iterations of the outermost loop (i.e., the loop on lines 2-17); let us consider the iteration processing $i \in [2, n]$ and let v be the new vertex in G_i. Let $\pi = (T, D, M, f)$ be an extended i-profile. Since $\sigma(\pi)$ has already been computed if $v \notin T$, it suffices to consider only profiles such that $v \in T$; these profiles can be enumerated in time $2^{\mathcal{O}(kd \log(kd))}$ by Lemma 6. There are at most $k + 1$ functions f' compatible with π and for each of them, we can find an f'-decomposition of size at most $m \leq d + 2$ in time $\mathcal{O}(k^2 d^2)$ by Lemma 8. Hence, the running time of one iteration of the outermost loop is $2^{\mathcal{O}(kd \log(kd))}$, which implies that the total running time is $2^{\mathcal{O}(kd \log(kd))} \cdot n$ as required. ◀

3.4 Partial Vertex Cover

In this section, we prove the following theorem.

▶ **Theorem 14.** *There is an algorithm that, given a graph G, a contraction sequence $(G_1 = G, \ldots, G_n)$ of width d, and two integers k and t, decides the* Partial Vertex Cover *problem in time $2^{\mathcal{O}(kd \log(kd))} \cdot n$.*

The proof of the theorem above is similar to the proof of Theorem 13. Hence, we first discuss informally the differences between the two proofs and only afterwards give the formal proof in Section 3.4.1. The first difference is already mentioned in Section 3.1; it suffices to consider extended i-profiles $\pi = (T, D, M, f)$ such that $M = \emptyset$.

Recall that the vertex-cover-value of a solution $S \subseteq \beta(T)$ is the number of edges in $G[\beta(T)]$ covered by S, see Definition 5. Hence, if $\mathcal{D} = \{(T_j, D_j, \emptyset, f_j) \colon j \in [m]\}$ is a decomposition of π, then the edges in $G[\beta(T_j)]$ covered by S are covered also by $S \cap \beta(T_j)$. This means that when merging solutions of profiles in $\mathcal{D}$, we must only add the number $\delta_{j\ell}$ of covered edges with one endpoint in $\beta(T_j)$ and the other in $\beta(T_\ell)$ for *distinct* $j, \ell \in [m]$. Since there cannot be a red edge connecting D_j and T_ℓ by definition of a basic decomposition, we can compute $\delta_{j\ell}$ by inspecting the *black* edges between T_j and T_ℓ. Observe that a black edge $uv \in B(G_{i-1})$ semi-induces a complete bipartite graph between $\beta(u)$ and $\beta(v)$ in G. Because we have to avoid double-counting, we obtain the following equation:

$$\delta_{j\ell} = \sum_{u \in T_j} \sum_{\substack{v \in T_\ell, \\ uv \in B(G_{i-1})}} f_j(u) \cdot |\beta(v)| + f_\ell(v) \cdot |\beta(u)| - f_j(u) \cdot f_\ell(v)$$

Using this equation, we can compute an optimal solution of π. The rest of the proof of Theorem 14 is analogous to the proof of Theorem 13.

3.4.1 The Details for Partial Vertex Cover

In this section, we show that Algorithm 2 solves the Partial Vertex Cover problem, thereby proving Theorem 14. We will reuse the notion of extended profiles, see Definition 4,

and the notion of solutions, see Definition 5. However, we will not need the third component of a profile, and so it will always be set to $\emptyset$, i.e., a profile will be a tuple $(T, D, \emptyset, f)$. Instead of the *vertex-cover-value* of a solution, we will say simply the *value*. We say that a solution is an *optimal* solution of π if no other solution of π has higher value than S.

Algorithm 2 Algorithm for Partial Vertex Cover

Input: A graph G, contraction sequence $(G_1 = G, \dots, G_n)$ of width d, and $k \in \mathbb{N}$.

1 **for each** *extended* 1*-profile* $\pi = (T, D, \emptyset, f)$ **do** $\sigma_1(\pi) := (D, 0)$
2 **for** i **from** 2 **to** n **do**
3 $v :=$ the new vertex in G_i
4 $u_1, u_2 :=$ the two vertices of G_{i-1} contracted into v
5 **for each** *extended* i*-profile* $\pi = (T, D, \emptyset, f)$ **do**
6 **if** $v \notin T$ **then** $\sigma_i(\pi) := \sigma_{i-1}(\pi)$; **continue** with next π
7 $\sigma_i(\pi) := (\mathsf{null}, -1)$
8 $T' := (T \setminus v) \cup \{u_1, u_2\}$
9 **for each** *function* $f' \colon T' \to [0, k]$ *compatible with* π **do**
10 Let $\{\pi_j \colon j \in [m]\}$ be an f'-decomposition of π s.t. $m \leq d + 2$; it exists by Lemma 8.
11 **for each** $j \in [m]$ **do**
12 $(T_j, D_j, \emptyset, f_j) := \pi_j$; $(S_j, \nu_j) := \sigma_{i-1}(\pi_j)$
13 **for each** $1 \leq j < \ell \leq m$ **do**
14 $\mathsf{value}_{j\ell} := 0$
15 **for each** $x \in T_j$, $y \in T_\ell$ *s.t.* $xy \in B(G_{i-1})$ **do**
16 $\mathsf{value}_{j\ell} := \mathsf{value}_{j\ell} + f_j(x) \cdot |\beta(y)| + f_\ell(y) \cdot |\beta(x)| - f_j(x) \cdot f_\ell(y)$
17 $S := \bigcup_{j \in [m]} S_j$; $\nu := \sum_{j \in [m]} \nu_j + \sum_{1 \leq j < \ell \leq m} \mathsf{value}_{j\ell}$
18 **if** $\nu \geq \nu_0$, *where* $(S_0, \nu_0) = \sigma_i(\pi)$ **then** $\sigma_i(\pi) := (S, \nu)$
19 $(S, \nu) := \sigma_n(\{u\}, \{u\}, \{u\}, \{(u, k)\})$, where $\{u\} := V(G_n)$
20 **return** $\nu \geq t$

We will proceed analogously to Section 3.3. The following lemma is the counterpart of Lemma 11.

▶ **Lemma 15.** *Let $i \in [2, n]$, $\pi = (T, D, \emptyset, f)$ be an extended i-profile, f' be a function compatible with π, and ν and S be the values computed by Algorithm 2 on line 17 in the iteration processing f'. If for each extended $(i-1)$-profile π', S' is a solution of π' of value ν', where (S', ν') is the pair computed in $\sigma_{i-1}(\pi')$, then S is a solution of π of value ν.*

Proof. Let $\{\pi_j \colon j \in [m]\}$ be the f'-decomposition of π found on line 10 in the iteration processing f', and for each $j \in [m]$, let T_j, D_j, f_j, S_j, and ν_j be the values computed on line 12. By Lemma 9, S is a solution of π. To simplify notation, we denote $\delta(A, B) = |\{uv \in E(G) \colon u \in \beta(A), v \in \beta(B), \{u, v\} \cap S \neq \emptyset\}|$ for any $A, B \subseteq V(G_i)$ or $A, B \subseteq V(G_{i-1})$. Hence, by Definition 5, we need to show that $\nu = \delta(T, T)$.

Observe that $\delta(T, T) = \sum_{1 \leq j \leq \ell \leq m} \delta(T_j, T_\ell)$. Since ν_j is the value of S_j and $S \cap T_j = S_j$, $\nu_j = \delta(T_j, T_j)$ for any $j \in [m]$. Let us fix $j, \ell \in [m]$ such that $j < \ell$. Now it suffices to show that $\mathsf{value}_{j\ell} = \delta(T_j, T_\ell)$ is true on line 17. Observe that $\delta(T_j, T_\ell) = \sum_{u \in T_j, v \in T_\ell} \delta(\{u\}, \{v\})$. Let us fix $u \in T_j$ and $v \in T_\ell$, and let $\delta(u, v) := \delta(\{u\}, \{v\})$. Suppose that $\delta(u, v) > 0$, which means that $u \in D_j$ or $v \in D_\ell$ because $S \cap \beta(\{u, v\}) \neq \emptyset$, see Definition 4. Observe that

$uv \notin R(G_{i-1})$ by condition 3 in the definition of a basic decomposition, see Section 2. Since $\delta(u,v) > 0$, we obtain $uv \in B(G_{i-1})$. Finally, observe that in the iteration of the loop on lines 15-16, $\delta(u,v)$ is added to $\mathsf{value}_{j\ell}$. Indeed, there are all possible edges between $\beta(u)$ and $\beta(v)$ in G, $f_j(u) = |S \cap \beta(u)|$ and $f_\ell(v) = |S \cap \beta(v)|$ by Definition 5, and the last term avoids double-counting the edges with both endpoints in S. We have shown that $\nu = \delta(T,T)$ as desired. ◀

The following lemma is the counterpart of Lemma 12 in the partial dominating set algorithm.

▶ **Lemma 16.** *For each $i \in [n]$ and each extended i-profile π, if the final value of $\sigma_i(\pi)$ computed by Algorithm 2 is (S,ν), then S is an optimal solution of π and ν is the value of S.*

Proof. Let $i \in [n]$ and $\pi = (T, D, \emptyset, f)$ be an extended i-profile. First, suppose that $i = 1$. Since $T = \{u\}$ for some $u \in V(G)$, there are no edges in $G[\beta(T)] = G[T]$, and the value of the only solution D of π is clearly 0, see line 1. Now suppose that $i > 1$. Let $v \in V(G_i)$ be the new vertex in G_i and let $u_1, u_2 \in V(G_{i-1})$ be the two vertices contracted into v. If $v \notin T$, then π is also an $(i-1)$-profile and, by induction hypothesis, $\sigma_i(\pi) = \sigma_{i-1}(\pi)$ is an optimal solution of π, see line 6. Hence, let us assume that $v \in T$ and let $T' = (T \setminus v) \cup \{u_1, u_2\}$. By induction hypothesis and Lemma 15, π has a solution since there is a function compatible with π.

Let S^* be an optimal solution of π, let ν^* be the value of S^*, and let $f' \colon T' \to [0,k]$ be defined as $f'(u) = |S^* \cap \beta(u)|$. As in the proof of Lemma 12, f' is compatible with π. Let us consider the iteration of the loop on lines 5-18 processing π and the iteration of the loop on lines 9-18 processing f'. Let $\{\pi_j \colon j \in [m]\}$ be the f'-decomposition of π found on line 10. For each $j \in [m]$, let T_j, D_j, f_j, S_j and ν_j be as in the algorithm. Let ν and S be the values computed on line 17. By Lemma 15, S is a solution of π with value ν. Similarly to the proof of Lemma 15, we denote $\delta(A,B) = |\{uv \in E(G) \colon u \in \beta(A), v \in \beta(B), \{u,v\} \cap S \neq \emptyset\}|$ and $\delta^*(A) = |\{uv \in E(G) \colon u \in \beta(A), v \in \beta(B), \{u,v\} \cap S^* \neq \emptyset\}|$ for any $A \subseteq V(G_i)$ or $A \subseteq V(G_{i-1})$.

Suppose for contradiction that $\nu < \nu^*$, i.e., $\delta(T,T) < \delta^*(T,T)$. Observe that $\delta(T,T) = \sum_{1 \le j \le \ell \le m} \delta(T_j, T_\ell)$ and $\delta^*(T,T) = \sum_{1 \le j \le \ell \le m} \delta^*(T_j, T_\ell)$. Hence, there are $j, \ell \in [m]$ such that $\delta(T_j, T_\ell) < \delta^*(T_j, T_\ell)$. Observe that for each $j, \ell \in [m]$, $j \neq \ell$, we have

$$\delta(T_j, T_\ell) = \sum_{\substack{u \in T_j, v \in T_\ell, \\ uv \in B(G_{i-1})}} f_j(u) \cdot |\beta(v)| + f_\ell(v) \cdot |\beta(u)| - f_j(u) \cdot f_\ell(v) = \delta^*(T_j, T_\ell).$$

Indeed, if $u \in T_j, v \in T_\ell$, and $uv \in R(G_{i-1})$, then $f_j(u) = 0 = f_\ell(v)$ by condition 3 in the definition of a basic decomposition, see Section 2. Hence, there is $j \in [m]$ such that $\delta(T_j, T_j) < \delta^*(T_j, T_j)$. However, $\delta(T_j, T_j)$ is the value of S_j and $\delta^*(T_j, T_j)$ is the value of $S^* \cap \beta(T_j)$, which is a solution of π_j. This is a contradiction since S_j is an optimal solution of π_j by induction hypothesis. Hence, $\nu = \nu^*$, and S is an optimal solution of π, too.

After the iteration of the loop on lines 9-18 processing f', $\sigma_i(\pi) = (S, \nu)$. Observe that the second component of $\sigma_i(\pi)$ cannot change in any of the following iterations because ν is always the value of some solution of π by Lemma 15. Hence, after all functions compatible with π have been processed, $\sigma_i(\pi)$ is indeed an optimal solution of π and its value, as desired. ◀

We are now ready to prove the main theorem of this section.

Proof of Theorem 14. Let $\iota = (G, C = (G = G_1, \dots, G_n), k, t)$ be the input instance, where C is contraction sequence of width d. Let $V(G_n) = \{u\}$ and let $\pi = (\{u\}, \{u\}, \emptyset, \{(u, k)\})$. Note that π is an extended n-profile unless $n < k$, in which case, (G, k, t) is a NO instance. By Lemma 16, S is an optimal solution of π with value ν, where (S, ν) is the pair computed by Algorithm 2. By Definition 5, $|S| = k$ and $\nu = |\{uv \in E(G) \colon u \in S, v \in \beta(\{u\})\}| = |\{uv \in E(G) \colon u \in S\}|$. Hence, (G, k, t) is a YES instance if and only if $\nu \geq t$.

The fact that Algorithm 2 achieves the desired running time can be proven analogously as for Algorithm 1, see the proof of Theorem 13. ◀

4 $(\exists^* \sum \#\text{QF})$Model Checking

We now turn our attention to the framework mentioned in the introduction. For a background in (counting) logic, we refer the readers to [31]. We consider a generalization of a fragment of the counting logic FO(>0), namely problems expressible by formulas of the form $\exists x_1 \cdots \exists x_k \sum_{\alpha \in I} \#y\, \psi_\alpha(x_1, \dots, x_k, y) \geq t$ for some quantifier-free formulas ψ_α, $\alpha \in I$, and $t \in \mathbb{N}$. Such a formula is true in a graph G if there are vertices $v_1, \dots, v_k \in V(G)$ such that $\sum_{\alpha \in I} \#y\, \psi_\alpha(v_1, \dots v_k, y) \geq t$, where $\#y\, \psi_\alpha(v_1, \dots, v_k, y)$ is the number of vertices $w \in V(G)$ such that $\psi_\alpha(v_1, \dots, v_k, w)$ holds in G.

We are interested in giving an FPT algorithm on graph classes of bounded twin-width to solve the model-checking problem for this logic.

$(\exists^* \sum \#\text{QF})$Model Checking using twin-width
Input: A graph G, a contraction sequence for G of width d, a finite set I, a quantifier-free formula ψ_α with $k+1$ free variables for each $\alpha \in I$, and $t \in \mathbb{N}$
Question: Are there vertices $v_1, \dots, v_k \in V(G)$ such that $G \models \sum_{\alpha \in I} \#y\, \psi_\alpha(v_1 \dots v_k, y) \geq t$?
Parameter: $k + d$[1]

It is not hard to express Partial Dominating Set in this logic, see Section 1. Let us illustrate how Partial Vertex Cover can be expressed.

▶ **Example 17.** We can express Partial Vertex Cover with $I = [k]$ and the following formulas. For $\alpha \in I$, let $\psi_\alpha(x_1, \dots, x_k, y) \equiv E(x_\alpha, y) \wedge \bigwedge_{\gamma \in [\alpha - 1]} y \neq x_\gamma$. Intuitively, ψ_α counts the number of edges covered by x_α. To avoid double-counting, the potential edge $x_\alpha x_\gamma$ is counted by ψ_α only if $\alpha < \gamma$ (and otherwise it is counted by ψ_γ).

4.1 Virtual Profiles

In Section 3, we used extended profiles, which were obtained by "extending" basic profiles introduced in Section 2. In our model checking algorithm, we will again "extend" the basic profiles in a certain way. This time, however, these profiles will need to contain even more information than the extended profiles. Intuitively, the reason behind this change is that the k existential variables in the considered formula are not interchangeable, whereas the k vertices in a partial dominating set (or vertex cover) are. Hence, a part of a virtual profile is a function f, which describes where each existential variable should be realized.

However, we need to be able to capture the fact that some variables may be realized in a part of G not covered by the profile; this is done by adding a bounded number of "virtual" vertices and edges ($\mathcal{V}$ and $\mathcal{E}$). Note that the virtual vertices are "new objects"; e.g.,

[1] Note that we assume that each formula ψ_α is normalized, i.e., there is no shorter formula equivalent to ψ_α. Without this assumption, the parameter would have to be $k + d + \sum_{\alpha \in I} |\psi_\alpha|$.

$V(G) \cap \mathcal{V} = \emptyset$. Moreover, we need to remember which atomic formulas ($x_a = x_b$, $E(x_a, x_b)$, and negations of these two) are satisfied by which existential variables: this information is encoded in a vertex-labeled graph H. Note that if a virtual profile has a solution, then H must be compatible with f and $\mathcal{E}$.

▶ **Definition 18.** Let $i \in [n]$ and let (T, D) be a basic i-profile. A *virtual i-profile* is a tuple $\pi = (T, D, \mathcal{V}, \mathcal{E}, f, H)$, where:

- $\mathcal{V}$ is the set of at most k *virtual* vertices.
- $\mathcal{E} \subseteq \{uv \colon u \in \mathcal{V} \cup T, v \in \mathcal{V}\}$ is the set of *virtual* edges.
- $f \colon [k] \to D \cup \mathcal{V}$ is a surjective function such that $|f^{-1}(u)| \leq |\beta(u)|$ for each $u \in D$.
- H is a graph with at most k vertices with vertex set labeled as $\{h_1, \ldots, h_k\}$.

The *expanded graph* of π, denoted G_π, is the graph with vertex set $\beta(T) \cup \mathcal{V}$ and edge set $E(G[\beta(T)]) \cup \{uv \in \mathcal{E} \colon u, v \in \mathcal{V}\} \cup \{u_0 v \colon uv \in \mathcal{E}, v \in \mathcal{V}, u \in T, u_0 \in \beta(u)\}$. A tuple $\mathbf{s} = (s_1, \ldots, s_k) \in V(G_\pi)^k$ is a *solution* of π if:

- for each $a \in [k]$, $s_a = f(a)$ if $f(a) \in \mathcal{V}$, and $s_a \in \beta(f(a))$ if $f(a) \in D$;
- the function $s_a \mapsto h_a$ is an isomorphism from $G_\pi[\{s_1, \ldots, s_k\}]$ to H.

The *value* of $\mathbf{s}$ is defined as $\sum_{\alpha \in I} |\{u \in \beta(T) \colon G_\pi \models \psi_\alpha(s_1, \ldots, s_k, u)\}|$. A solution $\mathbf{s}$ of π is *optimal* if no other solution of π has higher value than $\mathbf{s}$.

The expanded graph G_π illustrates the concept of virtual profiles: the subgraph of G induced by the bags from T is expanded with the virtual vertices. The virtual vertices are uniformly connected to vertices from each bag in T as dictated by $\mathcal{E}$. A solution $\mathbf{s}$ of π must then adhere to π in the following way: each s_a in $\mathbf{s}$ has to be the correct virtual vertex or a vertex from the correct bag as dictated by f; and $\mathbf{s}$ has to induce the desired subgraph H in the expanded graph G_π. When an optimal solution has been computed for each virtual profile, we can find the answer to the problem by inspecting an n-profile π^* such that $\mathcal{V}$ and $\mathcal{E}$ are empty, $T = V(G_n)$ and $\beta(T) = V(G)$, and f is the unique function from $[k]$ to $V(G_n)$. In contrast to the case of Partial Dominating Set, the profile π^* is not unique; one has to try all possible choices of H.

Now we show that the number of virtual profiles is bounded.

▶ **Lemma 19.** *If $i \in [n]$ and $v \in V(G_i)$, then the number of virtual i-profiles $(T, D, \mathcal{V}, \mathcal{E}, f, H)$ such that $v \in T$ is in $2^{\mathcal{O}(k^2 d \log(d))}$. Moreover, such profiles can be enumerated in time $2^{\mathcal{O}(k^2 d \log(d))}$.*

Proof. By Lemma 3, there is $d^{\mathcal{O}(kd)}$ basic i-profiles (T, D). Let us bound the number of possible graphs H. There are at most $k \cdot 2^{k^2}$ graphs on at most k vertices and for each of them, there are at most k^k vertex-labelings with labels $\{h_1, \ldots, h_k\}$. Hence, there are $2^{\mathcal{O}(k^2)}$ choices of H. There are $k+1$ choices of $\mathcal{V}$ (it suffices to choose the size of $\mathcal{V}$) and at most $2^{k \cdot (k + k(d+1))} \in 2^{\mathcal{O}(k^2 d)}$ choices of $\mathcal{E}$. Finally, there are at most $(k + k(d+1))^k \in 2^{\mathcal{O}(k \log(kd))}$ choices of f. Hence, there are $2^{\mathcal{O}(k^2 d \log(d))}$ virtual i-profiles containing v, and the claim about enumeration follows from Lemma 3. ◀

4.2 Virtual Decompositions

To achieve our goal of finding optimal solutions of virtual profiles, we continue similarly as in the previous section about Partial Dominating Set and Partial Vertex Cover. We will again need a notion of a decomposition for our virtual profiles. However, we must ensure that the profiles in the decomposition are "compatible" with each other with respect to the virtual vertices. Informally, the virtual edges must correspond to the black edges between the profiles in G_{i-1}.

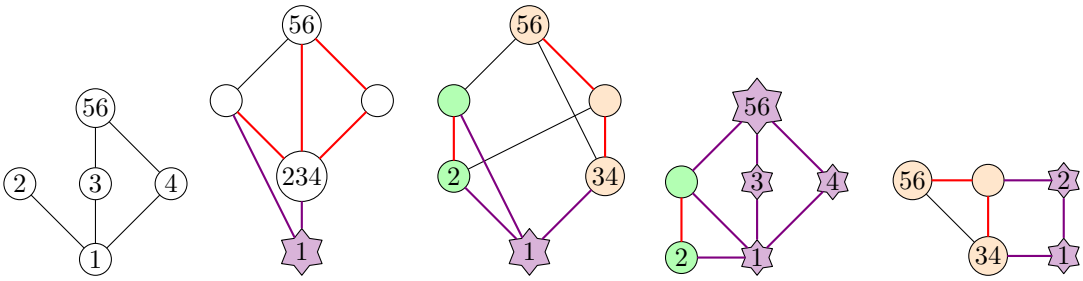

Figure 3 Leftmost: A graph H with vertex set $\{h_1, \dots, h_6\}$. The circle representing h_a contains the digit a. **Middle left:** A virtual i-profile $\pi = (T, D, \mathcal{V}, \mathcal{E}, f, H)$ for $k = 6$. Virtual vertices are represented by violet stars and virtual edges by violet lines. T is the set of all non-virtual vertices. For each $a \in [6]$, the vertex $f(a)$ contains the digit a. D is the set $f(\{2, 5\})$. **Middle:** The circles, and black and red edges represent $G_{i-1}[T']$, the violet star is the vertex in $\mathcal{V}$, and the violet lines represent $\mathcal{E}'$. The digits represent the function f'. The set T' has two red-connected components: T_1 colored in green and T_2 colored in orange. **Middle right:** The profile $(T_1, D_1, \mathcal{V}_1, \mathcal{E}_1, f_1, H)$. Note that the facts that $f_1(5) = f_1(6)$ and $f_1(3)f_1(4) \notin \mathcal{E}_1$ follow from H; they cannot be deduced from T or T'. **Rightmost:** The profile $(T_2, D_2, \mathcal{V}_2, \mathcal{E}_2, f_2, H)$.

Let $i \in [2, n]$, let $v \in V(G_i)$ be the new vertex in G_i, let $u_1, u_2 \in V(G_{i-1})$ be the two vertices contracted into v, let $\pi_0 = (T, D)$ be a basic i-profile such that $v \in T$, let $\pi = (T, D, \mathcal{V}, \mathcal{E}, f, H)$ be a virtual i-profile, let $T' = T \setminus \{v\} \cup \{u_1, u_2\}$, and let $\mathcal{E}' = \{uw \in \mathcal{E} \colon u, w \in T' \cup \mathcal{V}\} \cup \{u_1 w, u_2 w \colon vw \in \mathcal{E}\}$. We say that a function $f' \colon [k] \to T' \cup \mathcal{V}$ is *compatible* with π if $f'(a) \in \{u_1, u_2\}$ when $f(a) = v$, and $f'(a) = f(a)$ otherwise.

If a function f' is compatible with π, then a set $\{\pi_j = (T_j, D_j, \mathcal{V}_j, \mathcal{E}_j, f_j, H) \colon j \in [m]\}$ of virtual $(i-1)$-profiles is an *f'-decomposition* if $D' = \operatorname{range}(f') \cap T'$, $\{(T_1, D_1), \dots, (T_m, D_m)\}$ is a D'-decomposition of π_0, and for each $j \in [m]$:

- $\mathcal{V}_j = \mathcal{V} \cup \{h_a \colon f'(a) \in D' \setminus D_j\}$.
- $f_j(a) = f'(a)$ if $f'(a) \in D_j \cup \mathcal{V}$, and $f_j(a) = h_a$ if $f'(a) \in D' \setminus D_j$.
- $\mathcal{E}_j$ is the union of the following sets.
  - $\{uw \in \mathcal{E}' \colon u, w \in \mathcal{V} \cup T_j\}$.
  - $\{uh_a \colon u \in \mathcal{V} \cup T_j, f'(a) = w \in D' \setminus D_j, uw \in \mathcal{E}' \cup B(G_{i-1})\}$.
  - $\{h_a h_b \in E(H) \colon f'(a), f'(b) \in D' \setminus D_j\}$.

The intuition behind this construction of π_j is that for each vertex in $D' \setminus D_j$, which are the vertices with some variable assigned to them, we add a new virtual vertex to $\mathcal{V}_j$, and $\mathcal{E}_j$ contains $\mathcal{E}'$ "restricted" to T_j (the first term), edges between a new virtual vertex h_a and an "old" virtual vertex $u \in \mathcal{V}$ or a vertex $u \in T_j$ (the second term), and edges between two new virtual vertices (the third term). Note that we used the vertices of H to label the new virtual vertices, which means that two new virtual vertices h_a and h_b may be equal even when $a \neq b$; this ensures the possibility of an isomorphism from a solution of π_j to H. See Figure 3 for an illustration of a virtual decomposition.

Again, we prove that a small f'-decomposition exists for each f' compatible with π.

▶ **Lemma 20.** *If $i \in [2, n]$, $V(G_i) \setminus V(G_{i-1}) = \{v\}$, $V(G_{i-1}) \setminus V(G_i) = \{u_1, u_2\}$, $\pi = (T, D, \mathcal{V}, \mathcal{E}, f, H)$ is a virtual i-profile such that $v \in T$, $T' = T \setminus \{v\} \cup \{u_1, u_2\}$, and $f' \colon [k] \to T' \cup \mathcal{V}$ is a function compatible with π, then there is an f'-decomposition $\mathcal{D}$ of π with cardinality at most $d + 2$. Moreover, $\mathcal{D}$ can be found in time $\mathcal{O}(k^2 d^2)$.*

Proof. Let $D' = \operatorname{range}(f') \cap T'$. By Lemma 2, a D'-decomposition $\mathcal{D}_0 = \{(T_j, D_j) \colon j \in [m]\}$ of (T, D) such that $m \leq d + 2$ can be found in time $\mathcal{O}(kd^2)$. For each $j \in [m]$, define $\mathcal{V}_j$, $\mathcal{E}_j$,

and f_j as in the definition of an f'-decomposition above. Now it can be easily seen that $\{(T_j, D_j, \mathcal{V}_j, \mathcal{E}_j, f_j, H) \colon j \in [m]\}$ is the desired f'-decomposition of π.

Let $j \in [m]$ and observe that computing $\mathcal{V}_j$ and f_j is straightforward. To compute $\mathcal{E}_j$, we have to inspect all pairs $u \in T_j$ and $v \in D' \setminus D_j$. The number of all such pairs, for all $j \in [m]$ in total, is bounded by $|T'| \cdot |D'| \in \mathcal{O}(k^2 d)$. After adding the time required to find $\mathcal{D}_0$, we obtain the stated running time. ◀

4.3 Combining Solutions

In Section 3, we obtained a solution for an extended i-profile π by simply taking the union of solutions of profiles in a decomposition of π. In this section, combining solutions is more complicated because each solution is a tuple, which may contain some virtual vertices.

▶ **Definition 21.** Let $i \in [n]$, let $\pi_1 = (T_1, D_1, \mathcal{V}_1, \mathcal{E}_1, f_1, H)$ and $\pi_2 = (T_2, D_2, \mathcal{V}_2, \mathcal{E}_2, f_2, H)$ be two virtual i-profiles, and let $\mathbf{s}^1 = (s^1_1, \ldots, s^1_k)$ and $\mathbf{s}_2 = (s^2_1, \ldots, s^2_k)$ be solutions of π_1 and π_2, respectively. By $\mathbf{s}^1 \oplus \mathbf{s}^2$, we denote the tuple $(s_1, \ldots, s_k) \in (\beta(D_1) \cup V(G_{\pi_2}))^k$ such that for each $a \in [k]$, $s_a = s^1_a$ if $f_1(a) \in D_1$, and $s_a = s^2_a$ otherwise.

Note that in general, the operation $\oplus$ is neither associative nor commutative. However, we will use it only to combine solutions of profiles in a decomposition (and when restricted to such inputs, $\oplus$ is both associative and commutative, which will be implicit in the proof of the following lemma). Let $\mathbf{s} = \bigoplus_{j \in [m]} \mathbf{s}^j = \mathbf{s}^1 \oplus (\mathbf{s}^2 \oplus \ldots)$. Formally, $\mathbf{s} = \mathbf{s}^1$ if $m = 1$ and $\mathbf{s} = \mathbf{s}^1 \oplus \bigoplus_{j \in [2,m]} \mathbf{s}^j$ otherwise.

▶ **Lemma 22.** *If $i \in [2, n]$, π is a virtual i-profile, $\{\pi_j \colon j \in [m]\}$ is an f'-decomposition of π, and $\mathbf{s}^j$ is a solution of π_j for each $j \in [m]$, then $\mathbf{s} = \bigoplus_{j \in [m]} \mathbf{s}^j$ is a solution of π. Moreover, if ν_j is the value of $\mathbf{s}_j$, then the value of $\mathbf{s}$ is $\sum_{j \in [m]} \nu_j$.*

Proof. Let $\pi = (T, D, \mathcal{V}, \mathcal{E}, f, H)$ and $\mathbf{s} = (s_1, \ldots, s_k)$. For each $j \in [m]$, let $\pi_j = (T_j, D_j, \mathcal{V}_j, \mathcal{E}_j, f_j, H)$ and let $\mathbf{s}_j = (s^j_1, \ldots, s^j_k)$. For each $j \in [m]$, let $G^j := G_{\pi_j}$.

First, let $a \in [k]$ be such that $f(a) \in \mathcal{V}$. Observe that for each $j \in [m]$, $f(a) = f'(a) = f_j(a) = s^j_a$, which means that $s_a = f(a)$ as desired (because $s_a = s^j_a$ for some $j \in [m]$ by Definition 21). Second, let $a \in [k]$ be such that $f(a) \in D$ and let $j \in [m]$ be the unique index such that $u := f'(a) = f_j(a) \in D_j$. By Definition 21, $s_a = s^j_a \in \beta(u)$ because $f_\ell(a) \notin D_\ell$ for each $\ell \in [m] \setminus \{j\}$. By definition of f', if $u \in V(G_{i-1}) \cap V(G_i)$, then $f(a) = u$. Otherwise, $f(a)$ is the new vertex in G_i, and $s_a \in \beta(u) \subseteq \beta(f(a))$. In both cases, $s_a \in \beta(f(a))$ as desired.

Now we need to show that the function $s_a \mapsto h_a$ is an isomorphism from $G_\pi[\{s_1, \ldots, s_k\}]$ to H. Let us fix $a, b \in [k]$. Since $\mathbf{s}_j$ is a solution of π_j for each $j \in [m]$, it suffices to show that $s_a s_b \in E(G_\pi) \leftrightarrow s^j_a s^j_b \in E(G^j)$ and $s_a = s_b \leftrightarrow s^j_a = s^j_b$ for some $j \in [m]$. First, if $s_a, s_b \in \mathcal{V}$ or $s_a, s_b \in \beta(T_j)$ for some $j \in [m]$, then $s_a = s^j_a$ and $s_b = s^j_b$, and $s_a s_b \in E(G_\pi) \leftrightarrow s_a s_b \in E(G) \cup \mathcal{E} \leftrightarrow s^j_a s^j_b \in E(G^j)$ as required. Moreover, we clearly have $s_a = s_b \leftrightarrow s^j_a = s^j_b$; note that in all other cases, $s_a \neq s_b$ and $s^j_a \neq s^j_b$. Second, if $s_a \in \mathcal{V}$ and $s_b \in \beta(u) \subseteq \beta(u')$ for some $u \in T_j$, $j \in [m]$, and $u' \in T$, then $s_a s_b \in E(G_\pi) \leftrightarrow s_a u' \in \mathcal{E} \leftrightarrow s_a u \in \mathcal{E}_j \leftrightarrow s_a s_b \in E(G^j)$. Finally, assume that $j, \ell \in [m]$, $j \neq \ell$, and $s_a \in \beta(T_j), s_b \in \beta(T_\ell)$. Let $u = f'(a) = f_j(a)$ and $v = f'(b) = f_\ell(b)$. By definition of a solution, we have $s_a \in \beta(u)$ and $s_b \in \beta(v)$. Recall that $D' = \bigcup_{p \in [m]} D_p$, that $u, v \in D'$, and that $\{(T_p, D_p) \colon p \in [m]\}$ is a D'-decomposition of the basic i-profile (T, D). Hence, by condition 3 in the definition of a D'-decomposition, see Section 2, $uv \notin R(G_{i-1})$. Now $s_a s_b \in E(G_\pi) \leftrightarrow s_a s_b \in E(G) \leftrightarrow uv \in B(G_{i-1}) \leftrightarrow u h_b \in \mathcal{E}_j \leftrightarrow s^j_a s^j_b \in E(G^j)$ as required. Note that after the third equivalence, we could have used $h_a v \in \mathcal{E}_\ell$ and continued analogously. We have shown that $\mathbf{s}$ is a solution of π.

What remains to be shown is that the value of $\mathbf{s}$ equals $\sum_{j\in[m]}\nu_j$. Since $\beta(T)=\beta(T')$, where $T'=\bigcup_{j\in[m]}T_j$, it suffices to show that for each $j\in[m]$, $u\in\beta(T_j)$ and $\alpha\in I$, $G_\pi\models\psi_\alpha(s_1,\dots,s_k,u)$ if and only if $G^j\models\psi_\alpha(s_1^j,\dots,s_k^j,u)$. Using the isomorphisms to H, we obtain that $s_a\mapsto s_a^j$ is an isomorphism from $G_\pi[\{s_1,\dots,s_k\}]$ to $G^j[\{s_1^j,\dots,s_k^j\}]$. Hence, it suffices to show that for each $a\in[k]$, (1) $u=s_a\leftrightarrow u=s_a^j$, and (2) $us_a\in E(G_\pi)\leftrightarrow us_a^j\in E(G^j)$. If $s_a=s_a^j$, then (1) clearly holds, and if $s_a\neq s_a^j$, then $s_a\notin\beta(T_j)$ and $s_a^j\in\mathcal{V}_j$, which implies $u\notin\{s_a,s_a^j\}$. Now let us focus on (2). Let $u_j\in T_j$ and $u'\in T$ be such that $u\in\beta(u_j)\subseteq\beta(u')$. If $s_a=s_a^j\in\beta(T_j)$, then $us_a\in E(G_\pi)\leftrightarrow us_a\in E(G)\leftrightarrow us_a\in E(G^j)$ as required. If $s_a=s_a^j\notin\beta(T_j)$, then $s_a\in\mathcal{V}$, and $us_a\in E(G_\pi)\leftrightarrow u's_a\in\mathcal{E}\leftrightarrow u_js_a\in\mathcal{E}_j\leftrightarrow us_a\in E(G^j)$ as required. Finally, suppose $s_a\neq s_a^j$, i.e., $s_a\in\beta(T)\setminus\beta(T_j)$, and $s_a^j=h_a\in\mathcal{V}_j$. Let $w=f'(a)$ and observe that, by definition of a solution, $s_a\in\beta(w)$. Since $w\in D'$, we have $u_jw\notin R(G_{i-1})$, see condition 3 in the definition of a D'-decomposition. Hence, we have $us_a\in E(G_\pi)\leftrightarrow us_a\in E(G)\leftrightarrow u_jw\in B(G_{i-1})\leftrightarrow u_jh_a\in\mathcal{E}_j\leftrightarrow uh_a=us_a^j\in E(G^j)$ as required. We have proven that the value of $\mathbf{s}$ is $\sum_{j\in[m]}\nu_j$ as desired. ◀

4.4 Model Checking Algorithm

Algorithm 3 Algorithm for $(\exists^*\sum\#\mathrm{QF})$Model Checking

Input: A graph G, a contraction sequence for G of width d, a finite set I, a quantifier-free formula ψ_α with $k+1$ free variables for each $\alpha\in I$, and $t\in\mathbb{N}$

for each *virtual* 1*-profile* $\pi=(T,D,\mathcal{V},\mathcal{E},f,H)$ **do**
  $\mathbf{s}:=(f(1),\dots,f(k))$
  if $\mathbf{s}$ *is a solution of* π **then** $\sigma_1(\pi):=(\mathbf{s},\sum_{\alpha\in I}|\{y\in T\colon G_\pi\models\psi_\alpha(\mathbf{s},y)\}|)$
  else $\sigma_1(\pi):=(\mathsf{null},0)$
for i **from** 2 **to** n **do**
  $v:=$ the new vertex in G_i; $u_1,u_2:=$ the two vertices of G_{i-1} contracted into v
  for each *virtual* i*-profile* $\pi=(T,D,\mathcal{V},\mathcal{E},f,H)$ **do**
    if $v\notin T$ **then** $\sigma_i(\pi):=\sigma_{i-1}(\pi)$; **continue** with next π
    $\sigma_i(\pi):=(\mathsf{null},0)$
    $T':=(T\setminus v)\cup\{u_1,u_2\}$
    for each *function* $f'\colon[k]\to T'\cup\mathcal{V}$ *compatible with* π **do**
      Let $\{\pi_j\colon j\in[m]\}$ be an f'-decomposition of π s.t. $m\leq d+2$; it exists by Lemma 20.
      for each $j\in[m]$ **do**
        $(\mathbf{s}^j,\nu_j):=\sigma_{i-1}(\pi_j)$
        if $\mathbf{s}^j=$ *null* **then continue** with next f'
      $\nu:=\sum_{j\in[m]}\nu_j$; $\mathbf{s}:=\bigoplus_{j\in[m]}\mathbf{s}^j$
      if $\nu\geq\nu_0$, *where* $\sigma_i(\pi):=(\mathbf{s}_0,\nu_0)$ **then** $\sigma_i(\pi):=(\mathbf{s},\nu)$
solution := null; count := 0; $\{u\}:=V(G_n)$
for *each graph* H *with vertex set labeled as* $\{h_1,\dots,h_k\}$ **do**
  $(\mathbf{s},\nu):=\sigma_n(\{u\},\{u\},\emptyset,\emptyset,f,H)$, where $f(a)=u$ for each $a\in[k]$
  if $\nu\geq$ *count* **then** solution := $\mathbf{s}$; count := ν
return count $\geq t$

Now we can start proving the correctness of Algorithm 3. First, let us show that the

algorithm computes an optimal solution for each virtual profile π (or it discovers that π has no solution).

▶ **Lemma 23.** *For each $i \in [n]$ and each virtual i-profile π, if the final value of $\sigma_i(\pi)$ computed by Algorithm 3 is $(\mathbf{s}, \nu)$, then $\mathbf{s}$ is an optimal solution of π and ν is the value of $\mathbf{s}$. On the other hand, if $\sigma_i(\pi) = (\mathsf{null}, 0)$, then π has no solution.*

Proof. Let $i \in [n]$ and $\pi = (T, D, \mathcal{V}, \mathcal{E}, H)$ be a virtual i-profile. First, suppose that $i = 1$. Since $G = G_1$ has no red edges, $T = \{u\}$ for some $u \in V(G)$. Recall that $\beta(u) = \{u\}$. Hence, by Definition 18, $(f(1), \ldots, f(k))$ is the only possible solution of π, and its value is either 0 or 1, see line 3. Now suppose that $i > 1$. Let $v \in V(G_i)$ be the new vertex in G_i and let $u_1, u_2 \in V(G_{i-1})$ be the two vertices contracted into v. If $v \notin T$, then π is also an $(i-1)$-profile and, by induction hypothesis, $\sigma_i(\pi) = \sigma_{i-1}(\pi)$ is an optimal solution of π (or $(\mathsf{null}, 0)$), see line 8. Hence, let us assume that $v \in T$ and let $T' = (T \setminus v) \cup \{u_1, u_2\}$.

If π has no solution, then for each function f' compatible with π, the iteration of the loop on lines 11-17 processing f' must be exited on line 15; otherwise, a solution of π would be found on line 16 (by induction hypothesis and Lemma 22). Hence, $\sigma_i(\pi)$ always equals $(\mathsf{null}, 0)$ as required. Suppose that π has a solution. Let $\mathbf{s}^* = (s_1^*, \ldots, s_k^*)$ be an optimal solution of π, let ν^* be the value of S^*, and let $f' \colon [k] \to T' \cup \mathcal{V}$ be such that $f'(a) = s_a^*$ if $s_a^* \in \mathcal{V}$, and $s_a^* \in \beta(f'(a))$ otherwise. Observe that f' is compatible with π, and let us consider the iteration of the loop on lines 7-17 processing π and the iteration of the loop on lines 11-17 processing f'. Let $\{\pi_j \colon j \in [m]\}$ be the f'-decomposition of π found on line 12. For each $j \in [m]$, let $\pi_j = (T_j, D_j, \mathcal{V}_j, \mathcal{E}_j, H)$, let $\mathbf{s}_j$ and ν_j be as in the algorithm, and let ν and $\mathbf{s}$ be the values computed on line 16. By Lemma 22, ν is the value of $\mathbf{s}$.

For each $j \in [m]$, let $\mathbf{s}^{j*} = (s_1^{j*}, \ldots, s_k^{j*})$ be the solution of π_j such that for each $a \in [k]$, $s_a^{j*} = s_a^*$ if $s_a^* \in \beta(T_j) \cup \mathcal{V}$, and $s_a^{j*} = h_a$ otherwise. Observe that $\mathbf{s}^* = \bigoplus_{j \in [m]} \mathbf{s}^{j*}$, and so, by Lemma 22, $\nu^* = \sum_{j \in [m]} \nu_j^*$, where ν_j^* is the value of $\mathbf{s}^{j*}$. Suppose for contradiction that $\nu < \nu^*$. Since $\beta(T) = \beta(T')$ and $T' = \bigcup_{j \in [m]} T_j$, there is $j \in [m]$ such that $\nu_j < \nu_j^*$, which is a contradiction since $\mathbf{s}^j$ is an optimal solution of π_j by the induction hypothesis. Hence, $\mathbf{s}$ is an optimal solution of π, too.

After the iteration of the loop on lines 11-17 processing f', $\sigma_i(\pi) = (\mathbf{s}, \nu)$. Observe that the second component of $\sigma_i(\pi)$ cannot change in any of the following iterations because ν is always the value of some solution of π by Lemma 22. Hence, after all functions compatible with π have been processed, $\sigma_i(\pi)$ is indeed an optimal solution of π and its value, as desired. ◀

Using Lemma 23, it is easy to show the correctness of Algorithm 3.

▶ **Theorem 24.** *Algorithm 3 decides the $(\exists^* \sum \#\mathrm{QF})$Model Checking using twin-width problem in time $2^{\mathcal{O}(k^2 d \log(d))} n$.*

Proof. Suppose that the algorithm is given a graph G, a contraction sequence $(G_1 = G, \ldots, G_n)$ of width d, and a formula $\phi \equiv \exists x_1 \cdots \exists x_k \sum_{\alpha \in I} \#y\, \psi_\alpha(x_1, \ldots, x_k, y) \geq t$. Let us say that a graph H with vertex set labeled as $\{h_1, \ldots, h_k\}$ is a *template graph*. Let us consider an iteration of the loop on lines 19-21 processing a template graph H, let $\pi_H = (\{u\}, \{u\}, \emptyset, \emptyset, f, H)$ be as on line 20, and let $(\mathbf{s}_H, \nu_H)$ be the tuple computed on line 20. By Lemma 23, $\mathbf{s}_H$ is an optimal solution of π_H with value ν_H, or $(\mathbf{s}_H, \nu_H) = (\mathsf{null}, 0)$ if π_H has no solution. Since there are no virtual vertices in π_H, if π_H has a solution, then $\mathbf{s}_H$ is a tuple in $V(G)^k$, and $\nu_H = \sum_{\alpha \in I} |\{u \in V(G) \colon G \models \psi_\alpha(\mathbf{s}, u)\}|$.

Let $(\mathbf{s}, \nu)$ the final value of $(\mathsf{solution}, \mathsf{count})$ computed by Algorithm 3. If $\mathbf{s} = \mathsf{null}$ and $\nu = 0$, then π_H has no solution for any template graph H, which means that for any tuple

$\mathbf{s}' \in V(G)^k$, $\sum_{\alpha \in I} |\{u \in V(G) \colon G \models \psi_\alpha(\mathbf{s}', u)\}| = 0$, and (G, ϕ) is a YES-instance if and only if $t = 0$. Otherwise, $\mathbf{s} \in V(G)^k$, $\nu = \sum_{\alpha \in I} |\{u \in V(G) \colon G \models \psi_\alpha(\mathbf{s}, u)\}|$, and $\nu = \max\{\nu_H \colon H$ is a template graph$\}$. Clearly, for each tuple $(s_1, \ldots, s_k) \in V(G)^k$, there is a template graph H such that $s_a \mapsto h_a$ is an isomorphism from $G[\{s_1, \ldots, s_k\}]$ to H, which means that (G, ϕ) is a YES-instance if and only if $\nu \geq t$. This shows the correctness of Algorithm 3.

Now we need to argue that Algorithm 3 achieves the desired running time. As in the proof of Theorem 13, we maintain a single function σ (instead of σ_i for each $i \in [n]$). First, observe that by Lemma 19, there are at most $2^{\mathcal{O}(k^2 d \log(d))} \cdot n$ virtual 1-profiles. Hence, executing the for loop on lines 1-4 takes time at most $2^{\mathcal{O}(k^2 d \log(d))} \cdot n$. Second, observe that there are $n - 1$ iterations of the loop on lines 5-17; let us consider the iteration processing $i \in [2, n]$ and let v be the new vertex in G_i. Let $\pi = (T, D, \mathcal{V}, \mathcal{E}, f, H)$ be a virtual i-profile. Since $\sigma(\pi)$ has already been computed if $v \notin T$, it suffices to consider only profiles such that $v \in T$; these profiles can be enumerated in time $2^{\mathcal{O}(k^2 d \log(d))}$ by Lemma 19. There are at most 2^k functions f' compatible with π and for each of them, we can find an f'-decomposition of size at most $m \leq d + 2$ in time $\mathcal{O}(k^2 d^2)$ by Lemma 20. Hence, the running time of one iteration of the loop is $2^{\mathcal{O}(kd \log(kd))}$. Finally, since there are $2^{\mathcal{O}(k^2)}$ choices of H, executing the loop on lines 19-21 takes time $2^{\mathcal{O}(k^2)}$. In total, the running time is $2^{\mathcal{O}(k^2 d \log(d))} \cdot n$ as required. ◀

5 Conclusion

We have demonstrated that several optimization problems requiring counting can be efficiently solved on graphs with small twin-width. This encompasses not only well-known problems such as Partial Vertex Cover and Partial Dominating Set, but also all problems expressible by a restricted fragment of first-order counting logic. It would be desirable to extend this result by providing a more comprehensive fragment that accommodates additional problems. For bounded expansion, we can utilize formulas of the form $\exists x_1 \cdots \exists x_k \# y \, \phi(x_1, \ldots, x_k)$, where ϕ is an arbitrary first-order formula rather than a quantifier-free one, and even slightly more complex formulas [18, 34]. Can we achieve the same for graphs of bounded twin-width? Moreover, we can handle even more complicated formulas with nested counting quantifiers *approximately* [18]. Once again, does the same or a similar result hold true for bounded twin-width? It should be noted that the best result for nowhere-dense classes is basically the same as our result for twin-width [17].

References

1. Jochen Alber, Hans L. Bodlaender, Henning Fernau, Ton Kloks, and Rolf Niedermeier. Fixed parameter algorithms for DOMINATING SET and related problems on planar graphs. *Algorithmica*, 33(4):461–493, 2002. doi:10.1007/S00453-001-0116-5.
2. Hagit Attiya and Jennifer L. Welch. *Distributed computing - fundamentals, simulations, and advanced topics (2. ed.)*. Wiley series on parallel and distributed computing. Wiley, 2004.
3. Jakub Balabán, Daniel Mock, and Peter Rossmanith. Solving partial dominating set and related problems using twin-width. *CoRR*, abs/2504.18218, 2025. URL: https://doi.org/10.48550/arXiv.2504.18218, arXiv:2504.18218, doi:10.48550/ARXIV.2504.18218.
4. R. Balasubramanian, Michael R. Fellows, and Venkatesh Raman. An improved fixed-parameter algorithm for vertex cover. *Inf. Process. Lett.*, 65(3):163–168, 1998. doi:10.1016/S0020-0190(97)00213-5.
5. Édouard Bonnet, Colin Geniet, Eun Jung Kim, Stéphan Thomassé, and Rémi Watrigant. Twin-width III: max independent set, min dominating set, and coloring. *SIAM J. Comput.*, 53(5):1602–1640, 2024. doi:10.1137/21M142188X.
6. Édouard Bonnet, Eun Jung Kim, Stéphan Thomassé, and Rémi Watrigant. Twin-width I: tractable FO model checking. In Sandy Irani, editor, *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 601–612. IEEE, 2020. doi:10.1109/FOCS46700.2020.00062.
7. L. Sunil Chandran and Fabrizio Grandoni. Refined memorization for vertex cover. *Inf. Process. Lett.*, 93(3):123–131, 2005. doi:10.1016/J.IPL.2004.10.003.
8. Jianer Chen, Iyad A. Kanj, and Weijia Jia. Vertex cover: Further observations and further improvements. *J. Algorithms*, 41(2):280–301, 2001. doi:10.1006/JAGM.2001.1186.
9. Jianer Chen, Iyad A. Kanj, and Ge Xia. Improved upper bounds for vertex cover. *Theor. Comput. Sci.*, 411(40-42):3736–3756, 2010. doi:10.1016/J.TCS.2010.06.026.
10. Jianer Chen, Lihua Liu, and Weijia Jia. Improvement on vertex cover for low-degree graphs. *Networks*, 35(4):253–259, 2000. doi:10.1002/1097-0037(200007)35:4\%3C253::AID-NET3\%3E3.0.CO;2-K.
11. Anuj Dawar, Martin Grohe, and Stephan Kreutzer. Locally excluding a minor. In *22nd IEEE Symposium on Logic in Computer Science (LICS 2007), 10-12 July 2007, Wroclaw, Poland, Proceedings*, pages 270–279. IEEE Computer Society, 2007. doi:10.1109/LICS.2007.31.
12. Anuj Dawar and Stephan Kreutzer. Domination problems in nowhere-dense classes. In Ravi Kannan and K. Narayan Kumar, editors, *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2009, December 15-17, 2009, IIT Kanpur, India*, volume 4 of *LIPIcs*, pages 157–168. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2009. doi:10.4230/LIPIcs.FSTTCS.2009.2315.
13. Erik D. Demaine, Fedor V. Fomin, Mohammad Taghi Hajiaghayi, and Dimitrios M. Thilikos. Subexponential parameterized algorithms on bounded-genus graphs and H-minor-free graphs. *J. ACM*, 52(6):866–893, 2005. doi:10.1145/1101821.1101823.
14. Rodney G. Downey and Michael R. Fellows. Fixed parameter tractability and completeness III: some structural aspects of the W hierarchy. In Klaus Ambos-Spies, Steven Homer, and Uwe Schöning, editors, *Complexity Theory: Current Research, Dagstuhl Workshop, February 2-8, 1992*, pages 191–225. Cambridge University Press, 1992.
15. Rodney G. Downey and Michael R. Fellows. *Parameterized Complexity*. Monographs in Computer Science. Springer, 1999. doi:10.1007/978-1-4612-0515-9.
16. Jan Dreier, Ioannis Eleftheriadis, Nikolas Mählmann, Rose McCarty, Michal Pilipczuk, and Szymon Torunczyk. First-order model checking on monadically stable graph classes. In *65th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2024, Chicago, IL, USA, October 27-30, 2024*, pages 21–30. IEEE, 2024. doi:10.1109/FOCS61266.2024.00012.
17. Jan Dreier, Daniel Mock, and Peter Rossmanith. Evaluating restricted first-order counting properties on nowhere dense classes and beyond. In Inge Li Gørtz, Martin Farach-Colton, Simon J. Puglisi, and Grzegorz Herman, editors, *31st Annual European Symposium on*

Algorithms, ESA 2023, September 4-6, 2023, Amsterdam, The Netherlands, volume 274 of *LIPIcs*, pages 43:1–43:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICS.ESA.2023.43.

18 Jan Dreier and Peter Rossmanith. Approximate evaluation of first-order counting queries. In Dániel Marx, editor, *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pages 1720–1739. SIAM, 2021. doi:10.1137/1.9781611976465.104.

19 John A. Ellis, Hongbing Fan, and Michael R. Fellows. The dominating set problem is fixed parameter tractable for graphs of bounded genus. *J. Algorithms*, 52(2):152–168, 2004. doi:10.1016/J.JALGOR.2004.02.001.

20 Jörg Flum and Martin Grohe. Fixed-parameter tractability, definability, and model-checking. *SIAM J. Comput.*, 31(1):113–145, 2001. doi:10.1137/S0097539799360768.

21 Fedor V. Fomin, Petr A. Golovach, Tanmay Inamdar, and Tomohiro Koana. FPT approximation and subexponential algorithms for covering few or many edges. *Inf. Process. Lett.*, 185:106471, 2024. doi:10.1016/J.IPL.2024.106471.

22 Markus Frick and Martin Grohe. Deciding first-order properties of locally tree-decomposable structures. *J. ACM*, 48(6):1184–1206, 2001. doi:10.1145/504794.504798.

23 Jakub Gajarský, Michal Pilipczuk, Wojciech Przybyszewski, and Szymon Torunczyk. Twin-width and types. In Mikolaj Bojanczyk, Emanuela Merelli, and David P. Woodruff, editors, *49th International Colloquium on Automata, Languages, and Programming, ICALP 2022, July 4-8, 2022, Paris, France*, volume 229 of *LIPIcs*, pages 123:1–123:21. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPICS.ICALP.2022.123.

24 Petr A. Golovach and Yngve Villanger. Parameterized complexity for domination problems on degenerate graphs. In Hajo Broersma, Thomas Erlebach, Tom Friedetzky, and Daniël Paulusma, editors, *Graph-Theoretic Concepts in Computer Science, 34th International Workshop, WG 2008, Durham, UK, June 30 - July 2, 2008. Revised Papers*, volume 5344 of *Lecture Notes in Computer Science*, pages 195–205, 2008. doi:10.1007/978-3-540-92248-3_18.

25 Martin Grohe, Stephan Kreutzer, and Sebastian Siebertz. Deciding first-order properties of nowhere dense graphs. In David B. Shmoys, editor, *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 89–98. ACM, 2014. doi:10.1145/2591796.2591851.

26 Jiong Guo, Rolf Niedermeier, and Sebastian Wernicke. Parameterized complexity of vertex cover variants. *Theory Comput. Syst.*, 41:501–520, 10 2007. doi:10.1007/s00224-007-1309-3.

27 Mrinmoy Hota, Madhumangal Pal, and Tapan Kumar Pal. An efficient algorithm for finding a maximum weight k-independent set on trapezoid graphs. *Comput. Optim. Appl.*, 18(1):49–62, 2001. doi:10.1023/A:1008791627588.

28 Lawqueen Kanesh, Jayakrishnan Madathil, Sanjukta Roy, Abhishek Sahu, and Saket Saurabh. Further exploiting c-closure for FPT algorithms and kernels for domination problems. *SIAM J. Discret. Math.*, 37(4):2626–2669, 2023. doi:10.1137/22M1491721.

29 Joachim Kneis, Daniel Mölle, and Peter Rossmanith. Partial vs. complete domination: t-dominating set. In Jan van Leeuwen, Giuseppe F. Italiano, Wiebe van der Hoek, Christoph Meinel, Harald Sack, and Frantisek Plasil, editors, *SOFSEM 2007: Theory and Practice of Computer Science, 33rd Conference on Current Trends in Theory and Practice of Computer Science, Harrachov, Czech Republic, January 20-26, 2007, Proceedings*, volume 4362 of *Lecture Notes in Computer Science*, pages 367–376. Springer, 2007. doi:10.1007/978-3-540-69507-3_31.

30 Tomohiro Koana, Christian Komusiewicz, and Frank Sommer. Exploiting c-closure in kernelization algorithms for graph problems. *SIAM J. Discret. Math.*, 36(4):2798–2821, 2022. URL: https://doi.org/10.1137/21m1449476, doi:10.1137/21M1449476.

31 Dietrich Kuske and Nicole Schweikardt. First-order logic with counting. In *32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2017, Reykjavik, Iceland, June 20-23, 2017*, pages 1–12. IEEE Computer Society, 2017. doi:10.1109/LICS.2017.8005133.

32 Vahan Mkrtchyan and Garik Petrosyan. On the fixed-parameter tractability of the partial vertex cover problem with a matching constraint in edge-weighted bipartite graphs. *J. Graph Algorithms Appl.*, 26(1):91–110, 2022. doi:10.7155/JGAA.00584.

33 Vahan Mkrtchyan, Garik Petrosyan, K. Subramani, and Piotr Wojciechowski. On the partial vertex cover problem in bipartite graphs - a parameterized perspective. *Theory Comput. Syst.*, 68(1):122–143, 2024. doi:10.1007/S00224-023-10152-W.

34 Daniel Mock and Peter Rossmanith. Solving a family of multivariate optimization and decision problems on classes of bounded expansion. In Hans L. Bodlaender, editor, *19th Scandinavian Symposium and Workshops on Algorithm Theory, SWAT 2024, June 12-14, 2024, Helsinki, Finland*, volume 294 of *LIPIcs*, pages 35:1–35:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPICS.SWAT.2024.35.

35 Jaroslav Nesetril and Patrice Ossona de Mendez. *Sparsity - Graphs, Structures, and Algorithms*, volume 28 of *Algorithms and combinatorics*. Springer, 2012. doi:10.1007/978-3-642-27875-4.

36 Rolf Niedermeier and Peter Rossmanith. Upper bounds for vertex cover further improved. In Christoph Meinel and Sophie Tison, editors, *STACS 99, 16th Annual Symposium on Theoretical Aspects of Computer Science, Trier, Germany, March 4-6, 1999, Proceedings*, volume 1563 of *Lecture Notes in Computer Science*, pages 561–570. Springer, 1999. doi:10.1007/3-540-49116-3_53.

37 Rolf Niedermeier and Peter Rossmanith. On efficient fixed-parameter algorithms for weighted vertex cover. *J. Algorithms*, 47(2):63–77, 2003. doi:10.1016/S0196-6774(03)00005-1.

38 Geevarghese Philip, Venkatesh Raman, and Somnath Sikdar. Solving dominating set in larger classes of graphs: FPT algorithms and polynomial kernels. In Amos Fiat and Peter Sanders, editors, *Algorithms - ESA 2009, 17th Annual European Symposium, Copenhagen, Denmark, September 7-9, 2009. Proceedings*, volume 5757 of *Lecture Notes in Computer Science*, pages 694–705. Springer, 2009. doi:10.1007/978-3-642-04128-0_62.

39 Detlef Seese. Linear time computable problems and first-order descriptions. *Math. Struct. Comput. Sci.*, 6(6):505–526, 1996. doi:10.1017/s0960129500070079.

40 Ulrike Stege, Iris van Rooij, Alexander Hertel, and Philipp Hertel. An $o(pn + 1.151^p)$-algorithm for p-profit cover and its practical implications for vertex cover. In Prosenjit Bose and Pat Morin, editors, *Algorithms and Computation, 13th International Symposium, ISAAC 2002 Vancouver, BC, Canada, November 21-23, 2002, Proceedings*, volume 2518 of *Lecture Notes in Computer Science*, pages 249–261. Springer, 2002. doi:10.1007/3-540-36136-7_23.