

Explicit linear kernels via dynamic programming^{*} ^{**}

Valentin Garnero¹, Christophe Paul¹, Ignasi Sau¹, and Dimitrios M. Thilikos^{1,2}

¹ AIGCo project-team, CNRS, LIRMM, Montpellier, France.

² Department of Mathematics,
National and Kapodistrian University of Athens, Athens, Greece.

FirstName.FamilyName@lirmm.fr

Abstract. Several algorithmic meta-theorems on kernelization have appeared in the last years, starting with the result of Bodlaender *et al.* [FOCS 2009] on graphs of bounded genus, then generalized by Fomin *et al.* [SODA 2010] to graphs excluding a fixed minor, and by Kim *et al.* [ICALP 2013] to graphs excluding a fixed topological minor. Typically, these results guarantee the existence of linear or polynomial kernels on sparse graph classes for problems satisfying some generic conditions but, mainly due to their generality, it is not clear how to derive from them constructive kernels with explicit constants.

In this paper we make a step toward a fully constructive meta-kernelization theory on sparse graphs. Our approach is based on a more explicit protrusion replacement machinery that, instead of expressibility in CMSO logic, uses dynamic programming, which allows us to find an explicit upper bound on the size of the derived kernels. We demonstrate the usefulness of our techniques by providing the first explicit linear kernels for r -DOMINATING SET and r -SCATTERED SET on apex-minor-free graphs, and for PLANAR- \mathcal{F} -DELETION on graphs excluding a fixed (topological) minor in the case where all the graphs in \mathcal{F} are connected.

Keywords: parameterized complexity, linear kernels, dynamic programming, protrusion replacement, graph minors.

1 Introduction

Motivation. Parameterized complexity deals with problems whose instances I come equipped with an additional integer parameter k , and the objective is to obtain algorithms whose running time is of the form $f(k) \cdot \text{poly}(|I|)$, where f is some computable function (see [15, 16] for an introduction to the field). We will be only concerned with problems defined on graphs. A fundamental notion in parameterized complexity is that of *kernelization*, which asks for the existence of polynomial-time preprocessing algorithms that produce equivalent instances whose size depends exclusively (preferably polynomially or even linearly) on k . Finding kernels of size polynomial or linear in k (called *linear kernels*) is one of the major goals of this area.

An influential work in this direction was the linear kernel of Alber *et al.* [2] for DOMINATING SET on planar graphs, which was generalized by Guo and Niedermeier [19] to a

^{*} A short conference version of this article will appear in the *Proc. of the 31st Symposium on Theoretical Aspects of Computer Science (STACS)*, Lyon, France, March 2014.

^{**} This work was supported by the ANR project AGAPE (ANR-09-BLAN-0159) and the Languedoc-Roussillon Project “Chercheur d’avenir” KERNEL. The fourth author was co-financed by the E.U. (European Social Fund - ESF) and Greek national funds through the Operational Program “Education and Lifelong Learning” of the National Strategic Reference Framework (NSRF) - Research Funding Program: “Thales. Investing in knowledge society through the European Social Fund”.

family of problems on planar graphs. Several algorithmic meta-theorems on kernelization have appeared in the last years, starting with the result of Bodlaender *et al.* [7] on graphs of bounded genus. After that, similar results have been obtained on larger sparse graph classes, such as graphs excluding a minor [18] or a topological minor [22].

Typically, the above results guarantee the *existence* of linear or polynomial kernels on sparse graph classes for a number of problems satisfying some generic conditions but, mainly due to their generality, it is hard to derive from them *constructive* kernels with *explicit* constants. The main reason behind this non-constructibility is that the proofs rely on a property of problems called *Finite Integer Index* (FII) that, roughly speaking, allows to replace large “protrusions” (i.e., large subgraphs with small boundary to the rest of the graph) with “equivalent” subgraphs of constant size. This substitution procedure is known as *protrusion replacer*, and while its *existence* has been proved, so far, there is no generic way to *construct* it. Using the technology developed in [7], there are cases where protrusion replacements can become constructive given the expressibility of the problem in Counting Monadic Second Order (CMSO) logic. This approach is essentially based on extensions of Courcelle’s theorem [11] that, even when they offer constructibility, it is hard to extract from them any *explicit constant* that upper-bounds the size of the derived kernel.

Results and techniques. In this article we tackle the above issues and make a step toward a fully constructive meta-kernelization theory on sparse graphs with explicit constants. For this, we essentially substitute the algorithmic power of CMSO logic with that of dynamic programming on graphs of bounded decomposability (i.e., bounded treewidth). Our approach provides a dynamic programming framework able to construct a protrusion replacer for a wide variety of problems.

Loosely speaking, the framework that we present can be summarized as follows. First of all, we propose a general definition of a problem encoding for the tables of dynamic programming when solving parameterized problems on graphs of bounded treewidth. Under this setting, we provide general conditions on whether such an encoding can yield a protrusion replacer. While our framework can also be seen as a possible formalization of dynamic programming, our purpose is to use it for constructing protrusion replacement algorithms and linear kernels whose size is explicitly determined.

In order to obtain an explicit linear kernel for a problem Π , the main ingredient is to prove that when solving Π on graphs of bounded treewidth via dynamic programming, we can use tables such that the maximum difference between all the values that need to be stored is bounded by a function of the treewidth. For this, we prove in Theorem 1 that when the input graph excludes a fixed graph H as a (topological) minor, this condition is sufficient for constructing an explicit protrusion replacer algorithm, i.e., a polynomial-time algorithm that replaces a large protrusion with an equivalent one whose size can be bounded by an *explicit* constant. Such a protrusion replacer can then be used, for instance, whenever it is possible to compute a linear protrusion decomposition of the input graph (that is, an algorithm that partitions the graph into a part of size linear in $O(k)$ and a set of $O(k)$ protrusions). As there is a wealth of results for constructing such decompositions [7, 17, 18, 22], we can use them as a starting point and, by applying dynamic programming, obtain an explicit linear kernel for Π .

We demonstrate the usefulness of this general strategy by providing the first explicit linear kernels for three distinct families of problems on sparse graph classes. On the one hand, for each integer $r \geq 1$, we provide a linear kernel for r -DOMINATING SET and r -SCATTERED SET on graphs excluding a fixed apex graph H as a minor. Moreover, for each finite family \mathcal{F} of connected graphs containing at least one planar graph, we provide a linear kernel for PLANAR- \mathcal{F} -DELETION on graphs excluding a fixed graph H

as a (topological) minor¹. We chose these families of problems as they are all tuned by a secondary parameter that is either the constant r or the size of the graphs in the family \mathcal{F} . That way, we not only capture a wealth of parameterized problems, but we also make explicit the contribution of the secondary parameter in the size of the derived kernels.

Organization of the paper. For the reader not familiar with the background used in previous work on this topic [7, 18, 22], some preliminaries can be found in Section 2, including graph minors, parameterized problems, (rooted) tree-decompositions, bounded graphs, the canonical equivalence relation $\equiv_{\Pi, t}$ for a problem Π and an integer t , FII, protrusions, and protrusion decompositions. In Section 3 we introduce the basic definitions of our framework and present an explicit protrusion replacer. The next three sections are devoted to showing how to apply our methodology to various families of problems, namely, we focus on r -DOMINATING SET in Section 4, on r -SCATTERED SET in Section 5, and on PLANAR- \mathcal{F} -DELETION in Section 6. Finally, we conclude with some directions for further research in Section 7.

2 Preliminaries

Graphs and minors. We use standard graph-theoretic notation (see [14] for any undefined terminology). Given a graph G , we let $V(G)$ denote its vertex set and $E(G)$ its edge set. For $X \subseteq V(G)$, we let $G[X]$ denote the graph (X, E_X) , where $E_X := \{xy \mid x, y \in X \text{ and } xy \in E(G)\}$, and we define $G - X := G[V(G) \setminus X]$. The open (resp. closed) *neighborhood* of a vertex v is denoted by $N(v)$ (resp. $N[v]$), and more generally, for an integer $r \geq 1$, we denote by $N_r(v)$ the set of vertices that are at distance at most r from v . The neighborhoods of a set of vertices S are defined analogously. The *distance* between a vertex v and a set of vertices S is defined as $d(v, S) = \min_{u \in S} d(v, u)$, where $d(v, u)$ denotes the usual distance. A graph $G = (E, V)$ is an *apex graph* if there exists $v \in V$ such that $G - v$ is planar. Given an edge $e = xy$ of a graph G , we let G/e denote the graph obtained from G by *contracting* the edge e , which amounts to deleting the endpoints of e , introducing a new vertex v_{xy} , and making it adjacent to all vertices in $(N(x) \cup N(y)) \setminus \{x, y\}$. A *minor* of G is a graph obtained from a subgraph of G by contracting zero or more edges. A *topological minor* of G is a graph obtained from a subgraph of G by contracting zero or more edges, such that each edge that is contracted has at least one endpoint with degree at most two. A graph G is H -(topological)-minor-free if G does not contain H as a (topological) minor.

Parameterized problems, kernels and treewidth. A *parameterized graph problem* Π is a set of pairs (G, k) , where G is a graph and $k \in \mathbb{Z}$, such that for any two instances (G_1, k_1) and (G_2, k_2) with $k_1, k_2 < 0$ it holds that $(G_1, k_1) \in \Pi$ if and only if $(G_2, k_2) \in \Pi$. If \mathcal{G} is a graph class, we define Π *restricted to* \mathcal{G} as $\Pi_{\mathcal{G}} = \{(G, k) \mid (G, k) \in \Pi \text{ and } G \in \mathcal{G}\}$. A *kernelization algorithm*, or just *kernel*, for a parameterized graph problem Π is an algorithm that given an instance (G, k) outputs, in time polynomial in $|G| + k$, an instance (G', k') of Π such that $(G, k) \in \Pi$ if and only if $(G', k') \in \Pi$ and $|G'|, k' \leq g(k)$, where g is some computable function. The function g is called the *size* of the kernel. If $g(k) = k^{O(1)}$ or $g(k) = O(k)$, we say that Π admits a *polynomial kernel* and a *linear kernel*, respectively.

¹ In an earlier version of this paper, we also described a linear kernel for PLANAR- \mathcal{F} -PACKING on graphs excluding a fixed graph H as a minor. Nevertheless, as this problem is not directly vertex-certifiable (see Definition 5), for presenting it we should restate and extend many of the definitions and results given in Section 3 in order to deal with more general families of problems. Therefore, we decided not to include this family of problems in this article.

Given a graph $G = (V, E)$, a *tree-decomposition* of G is an ordered pair $(T, \mathcal{X} = \{B_x \mid x \in V(T)\})$, where T is a tree and such that the following hold:

- (i) $\bigcup_{x \in V(T)} B_x = V(G)$;
- (ii) for every edge $e = uv$ in G , there exists $x \in V(T)$ such that $u, v \in B_x$; and
- (iii) for each vertex $u \in V(G)$, the set of nodes $\{x \in V(T) \mid u \in B_x\}$ induces a subtree.

The vertices of the tree T are usually referred to as *nodes* and the sets B_x are called *bags*. The *width* of a tree-decomposition is the size of a largest bag minus one. The *treewidth* of G , denoted $\text{tw}(G)$, is the smallest width of a tree-decomposition of G . A *rooted tree-decomposition* is a tree-decomposition $(T, \mathcal{X} = \{B_x \mid x \in V(T)\})$ in which a distinguished node $r \in V(T)$ has been selected as the *root*. The bag B_r is called the *root-bag*. Note that the root defines a child/parent relation between every pair of adjacent nodes in T , and ancestors/descendants in the usual way. A node without children is called a *leaf*.

For the definition of *nice tree-decompositions*, we refer to [23]. A set of vertices X of a graph G is called a *treewidth-modulator* if $\text{tw}(G - X) \leq t$, where t is some fixed constant.

Given a bag B of a tree-decomposition with tree T , we denote by T_B the subtree rooted at the node corresponding to bag B , and by $G_B := G[\bigcup_{x \in T_B} B_x]$ the subgraph of G induced by the vertices appearing in the bags corresponding to the nodes of T_B . If a bag B is associated with a node x of T , we may interchangeably use G_B or G_x .

Boundaried graphs and canonical equivalence relation. The following two definitions are taken from [7].

Definition 1 (Boundaried graphs). A boundaried graph is a graph G with a set $B \subseteq V(G)$ of distinguished vertices and an injective labeling $\lambda : B \rightarrow \mathbb{N}^+$. The set B is called the *boundary* of G and the vertices in B are called *boundary vertices*. Given a boundaried graph G , we denote its boundary by $\partial(G)$, we denote its labeling by λ_G , and we define its label set by $\Lambda(G) = \{\lambda_G(v) \mid v \in \partial(G)\}$. We say that a boundaried graph is a t -boundaried graph if $\Lambda(G) \subseteq \{1, \dots, t\}$.

Note that a 0-boundaried graph is just a graph with no boundary.

Definition 2 (Gluing operation). Let G_1 and G_2 be two boundaried graphs. We denote by $G_1 \oplus G_2$ the graph obtained by taking the disjoint union of G_1 and G_2 and identifying vertices with the same label of the boundaries of G_1 and G_2 . In $G_1 \oplus G_2$ there is an edge between two labeled vertices if there is an edge between them in G_1 or in G_2 .

Following [7], we introduce a canonical equivalence relation on boundaried graphs.

Definition 3 (Canonical equivalence on boundaried graphs). Let Π be a parameterized graph problem and let $t \in \mathbb{N}^+$. Given two t -boundaried graphs G_1 and G_2 , we say that $G_1 \equiv_{\Pi, t} G_2$ if $\Lambda(G_1) = \Lambda(G_2)$ and there exists a transposition constant $\Delta_{\Pi, t}(G_1, G_2) \in \mathbb{Z}$ such that for every t -boundaried graph $H \in \mathcal{G}$ and every $k \in \mathbb{Z}$, it holds that $(G_1 \oplus H, k) \in \Pi$ if and only if $(G_2 \oplus H, k + \Delta_{\Pi, t}(G_1, G_2)) \in \Pi$.

We define in Section 3 another equivalence relation on boundaried graphs that refines this canonical one (cf. Definitions 9 and 10), and that will allow us to perform a constructive protrusion replacement with explicit bounds.

The notion of *Finite Integer Index* was originally defined by Bodlaender and van Antwerpen-de Fluiter [9, 28]. We would like to note that FII does not play any role in the framework that we present for constructing explicit kernels, but we present its definition for completeness, as we will sometimes refer to it throughout the article.

Definition 4 (Finite Integer Index (FII)). *A parameterized graph problem Π has Finite Integer Index (FII for short) if for every positive integer t , the equivalence relation $\equiv_{\Pi,t}$ has finite index.*

Protrusions and protrusion decompositions. Given a graph $G = (V, E)$ and a set $W \subseteq V$, we define $\mathbf{bd}(W)$ as the vertices in W that have a neighbor in $V \setminus W$. A set $W \subseteq V(G)$ is a t -protrusion if $|\mathbf{bd}(W)| \leq t$ and $\mathbf{tw}(G[W]) \leq t - 1$. We would like to note that a t -protrusion W can be naturally seen as a t -boundaried graph by arbitrarily assigning labels to the vertices in $\mathbf{bd}(W)$. In this case, it clearly holds that $\partial(W) = \mathbf{bd}(W)$. Note also that if G is a t -boundaried graph of treewidth at most $t - 1$, we may assume that the boundary vertices are contained in any specified bag of a tree-decomposition, by increasing the width of the given tree-decomposition to at most $2t - 1$.

An (α, t) -protrusion decomposition of a graph G is a partition $\mathcal{P} = Y_0 \uplus Y_1 \uplus \dots \uplus Y_\ell$ of $V(G)$ such that:

- (i) for every $1 \leq i \leq \ell$, $N(Y_i) \subseteq Y_0$;
- (ii) $\max\{\ell, |Y_0|\} \leq \alpha$; and
- (iii) for every $1 \leq i \leq \ell$, $Y_i \cup N_{Y_0}(Y_i)$ is a t -protrusion of G .

When G is the input of a parameterized graph problem with parameter k , we say that an (α, t) -protrusion decomposition of G is *linear* whenever $\alpha = O(k)$.

3 An explicit protrusion replacer

In this section we present our strategy to construct an explicit protrusion replacer via dynamic programming. For a positive integer t , we define \mathcal{F}_t as the class of all t -boundaried graphs of treewidth at most $t - 1$ that have a rooted tree-decomposition with all boundary vertices contained in the root-bag. We will restrict ourselves to parameterized graph problems such that a solution can be certified by a subset of vertices.

Definition 5 (Vertex-certifiable problem). *A parameterized graph problem Π is called vertex-certifiable if there exists a language L_Π (called certifying language for Π) defined on pairs (G, S) , where G is a graph and $S \subseteq V(G)$, such that (G, k) is a YES-instance of Π if and only if there exists a subset $S \subseteq V(G)$ with $|S| \leq k$ (or $|S| \geq k$, depending on the problem) such that $(G, S) \in L_\Pi$.*

Many graph problems are vertex-certifiable, like r -DOMINATING SET, FEEDBACK VERTEX SET, or TREewidth- t VERTEX DELETION. This section is structured as follows. In Subsection 3.1 we define the notion of encoder, the main object that will allow us to formalize in an abstract way the tables of dynamic programming. In Subsection 3.2 we use encoders to define an equivalence relation on graphs in \mathcal{F}_t that, under some natural technical conditions, will be a *refinement* of the canonical equivalence relation defined by a problem Π (see Definition 3 in Section 2). This refined equivalence relation allows us to provide an explicit upper bound on the size of its representatives (Lemma 3), as well as a linear-time algorithm to find them (Lemma 4). In Subsection 3.3 we use the previous ingredients to present an explicit protrusion replacement rule (Theorem 1), which replaces a large enough protrusion with a bounded-size representative from its equivalence class, in such a way that the parameter does not increase.

3.1 Encoders

The DOMINATING SET problem, as a vertex-certifiable problem, will be used hereafter as a running example to illustrate our general framework and definitions. Let us start with a description of dynamic programming tables for DOMINATING SET on graphs of bounded treewidth.

Running example: Let B be a bag of a rooted tree-decomposition (T, \mathcal{X}) of width $t - 1$ of a graph $G \in \mathcal{F}_t$. The dynamic programming (DP) tables for DOMINATING SET can be defined as follows. The entries of the DP-table for B are indexed by the set of tuples $R \in \{0, \uparrow 1, \downarrow 1\}^{|B|}$, so-called *encodings*. As detailed below, the symbol 0 stands for vertices in the (partial) dominating set, the symbol $\downarrow 1$ for vertices that are already dominated, and $\uparrow 1$ for vertices with no constraints. More precisely, the coordinates of each $|B|$ -tuple are in one-to-one correspondence with the vertices of B . For a vertex $v \in B$, we denote by $R(v)$ its corresponding coordinate in the encoding R . A subset $S \subseteq V(G_B)$ is a *partial dominating set satisfying R* if the following conditions are satisfied:

- $\forall v \in V(G_B) \setminus B, d_{G_B}(v, S) \leq 1$; and
- $\forall v \in B: R(v) = 0 \Rightarrow v \in S$, and $R(v) = \downarrow 1 \Rightarrow d_{G_B}(v, S) \leq 1$.

Observe that if S is a partial dominating set satisfying R , then $\{v \in B \mid R(v) = 0\} \subseteq S$, but S may also contain vertices with $R(v) \neq 0$. Likewise, the vertices that are not (yet) dominated by S are contained in the set $\{v \in B \mid R(v) = \uparrow 1\}$. \diamond

The following definition considers the tables of dynamic programming in an abstract way.

Definition 6 (Encoder). An encoder \mathcal{E} is a pair $(\mathcal{C}, L_{\mathcal{C}})$ where

- \mathcal{C} is a function that, for each (possibly empty) finite subset $I \subseteq \mathbb{N}^+$, outputs a (possibly empty) finite set $\mathcal{C}(I)$ of strings over some alphabet. Each $R \in \mathcal{C}(I)$ is called a \mathcal{C} -encoding of I ; and
- $L_{\mathcal{C}}$ is a computable language whose strings encode triples (G, S, R) , where G is a boundaried graph, $S \subseteq V(G)$, and $R \in \mathcal{C}(\Lambda(G))$. If $(G, S, R) \in L_{\mathcal{C}}$, we say that S satisfies the \mathcal{C} -encoding R .

As it will become clear with the running example, the set I represents the labels from a bag, $\mathcal{C}(I)$ represents the possible configurations of the vertices in the bag, and $L_{\mathcal{C}}$ contains triples that correspond to solutions to these configurations.

Running example: Each rooted graph G_B can be naturally viewed as a $|B|$ -boundaried graph such that $B = \partial(G_B)$ with $I = \Lambda(G_B)$. Let $\mathcal{E}_{\text{DS}} = (\mathcal{C}_{\text{DS}}, L_{\mathcal{C}_{\text{DS}}})$ be the encoder described above for DOMINATING SET. The tables of the dynamic programming algorithm to solve DOMINATING SET are obtained by assigning to every \mathcal{C}_{DS} -encoding (that is, DP-table entry) $R \in \mathcal{C}_{\text{DS}}(I)$, the size of a minimum partial dominating set satisfying R , or $+\infty$ if such a set of vertices does not exist. This defines a function $f_G^{\mathcal{E}_{\text{DS}}} : \mathcal{C}_{\text{DS}}(I) \rightarrow \mathbb{N} \cup \{+\infty\}$. Observe that if $B = \partial(G_B) = \emptyset$, then the value assigned to the encodings in $\mathcal{C}_{\text{DS}}(\emptyset)$ is indeed the size of a minimum dominating set of G_B . \diamond

In the remainder of this subsection we will state several definitions for minimization problems, and we will restate them for maximization problems whenever some change is needed. For a general minimization problem Π , we will only be interested in encoders that permit to solve Π via dynamic programming. More formally, we define a Π -encoder and the values assigned to the encodings as follows.

Definition 7 (Π -encoder and its associated function). Let Π be a vertex-certifiable minimization problem.

- (i) An encoder $\mathcal{E} = (\mathcal{C}, L_{\mathcal{C}})$ is a Π -encoder if $\mathcal{C}(\emptyset)$ consists of a single \mathcal{C} -encoding, namely R_{\emptyset} , such that for every 0-boundaried graph G and every $S \subseteq V(G)$, $(G, S, R_{\emptyset}) \in L_{\mathcal{C}}$ if and only if $(G, S) \in L_{\Pi}$.
- (ii) Let G be a t -boundaried graph with $\Lambda(G) = I$. We define the function $f_G^{\mathcal{E}} : \mathcal{C}(I) \rightarrow \mathbb{N} \cup \{+\infty\}$ as

$$f_G^{\mathcal{E}}(R) = \min\{k : \exists S \subseteq V(G), |S| \leq k, (G, S, R) \in L_{\mathcal{C}}\}. \quad (1)$$

In Equation (1), if such a set S does not exist, we set $f_G^{\mathcal{E}}(R) := +\infty$. We define $\mathcal{C}_G^*(I) := \{R \in \mathcal{C}(I) \mid f_G^{\mathcal{E}}(R) \neq +\infty\}$.

Condition (i) in Definition 7 guarantees that, when the considered graph G has no boundary, the language of the encoder is able to *certify* a solution of problem Π . In other words, we ask that the set $\{(G, S) \mid (G, S, R_{\emptyset}) \in L_{\mathcal{C}}\}$ is a *certifying language* for Π . Observe that for a 0-boundaried graph G , the function $f_G^{\mathcal{E}}(R_{\emptyset})$ outputs the minimum size of a set S such that $(G, S) \in L_{\Pi}$.

For encoders $\mathcal{E}' = (\mathcal{C}', L_{\mathcal{C}'})$ that will be associated with problems where the objective is to find a set of vertices of size at least some value, the corresponding function $f_G^{\mathcal{E}'} : \mathcal{C}'(I) \rightarrow \mathbb{N} \cup \{-\infty\}$ is defined as

$$f_G^{\mathcal{E}'}(R) = \max\{k : \exists S \subseteq V(G), |S| \geq k, (G, S, R) \in L_{\mathcal{C}'}\}. \quad (2)$$

Similarly, in Equation (2), if such a set S does not exist, we set $f_G^{\mathcal{E}'}(R) := -\infty$. We define $\mathcal{C}_G^*(I) := \{R \in \mathcal{C}(I) \mid f_G^{\mathcal{E}}(R) \neq -\infty\}$.

The following definition provides a way to control the number of possible distinct values assigned to encodings. This property will play a similar role to FII or *monotonicity* in previous work [7, 18, 22].

Definition 8 (Confined encoding). An encoder \mathcal{E} is g -confined if there exists a function $g : \mathbb{N} \rightarrow \mathbb{N}$ such that for any t -boundaried graph G with $\Lambda(G) = I$ it holds that either $\mathcal{C}_G^*(I) = \emptyset$ or

$$\max_{R \in \mathcal{C}_G^*(I)} f_G^{\mathcal{E}}(R) - \min_{R \in \mathcal{C}_G^*(I)} f_G^{\mathcal{E}}(R) \leq g(t). \quad (3)$$

See Fig. 1 for a schematic illustration of a confined encoder. In this figure, each column of the table corresponds to a \mathcal{C} -encoder R , which is filled with the value $f_G^{\mathcal{E}}(R)$.

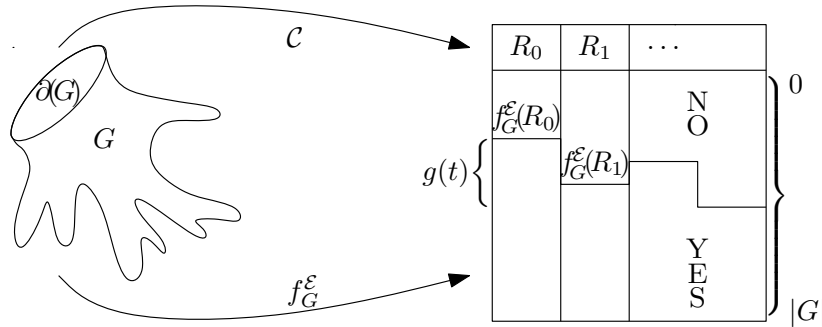


Fig. 1. Schematic illustration of a g -confined encoding.

Running example: It is easy to observe that the encoder \mathcal{E}_{DS} described above is g -confined for $g(t) = t$. Indeed, let G be a t -boundaried graph (corresponding to the graph G_B considered before) with $\Lambda(G) = I$. Consider an arbitrary encoding $R \in \mathcal{C}(I)$ and the encoding $R_0 \in \mathcal{C}(I)$ satisfying $R_0(v) = 0$ for every $v \in \partial(G)$. Let $S_0 \subseteq V(G)$ be a minimum-sized partial dominating set satisfying R_0 , i.e., such that $(G, S_0, R_0) \in L_{\mathcal{C}_{\text{DS}}}$. Observe that S_0 also satisfies R , i.e., $(G, S_0, R) \in L_{\mathcal{C}_{\text{DS}}}$. It then follows that $f_G^{\mathcal{E}_{\text{DS}}}(R_0) = \max_R f_G^{\mathcal{E}_{\text{DS}}}(R)$. Moreover, let $S \subseteq V(G)$ be a minimum-sized partial dominating set satisfying R , i.e., such that $(G, S, R) \in L_{\mathcal{C}_{\text{DS}}}$. Then note that R_0 is satisfied by the set $S \cup \partial(G)$, so we have that for every encoding R , $f_G^{\mathcal{E}_{\text{DS}}}(R) + |\partial(G)| \geq f_G^{\mathcal{E}_{\text{DS}}}(R_0)$. It follows that $f_G^{\mathcal{E}_{\text{DS}}}(R_0) - \min_R f_G^{\mathcal{E}_{\text{DS}}}(R) \leq |\partial(G)| \leq t$, proving that the encoder is indeed g -confined. \diamond

For some problems and encoders, we may need to “force” the confinement of an encoder \mathcal{E} that may not be confined according to Definition 8, while still preserving its usefulness for dynamic programming, in the sense that no relevant information is removed from the tables (for example, see the encoder for r -SCATTERED SET in Subsection 5.1). To this end, given a function $g : \mathbb{N} \rightarrow \mathbb{N}$, we define the function $f_G^{\mathcal{E},g} : \mathcal{C}(I) \rightarrow \mathbb{N} \cup \{+\infty\}$ as

$$f_G^{\mathcal{E},g}(R) = \begin{cases} +\infty, & \text{if } f_G^{\mathcal{E}}(R) - g(t) > \min_{R \in \mathcal{C}(I)} f_G^{\mathcal{E}}(R) \\ f_G^{\mathcal{E}}(R), & \text{otherwise.} \end{cases} \quad (4)$$

Intuitively, one shall think as the function $f_G^{\mathcal{E},g}$ as a “compressed” version of the function $f_G^{\mathcal{E}}$, which stores only the values that are useful for performing dynamic programming.

For encoders $\mathcal{E}' = (\mathcal{C}', L_{\mathcal{C}'})$ associated with maximization problems, given a function $g : \mathbb{N} \rightarrow \mathbb{N}$, we define the function $f_G^{\mathcal{E}',g} : \mathcal{C}(I) \rightarrow \mathbb{N} \cup \{-\infty\}$ as

$$f_G^{\mathcal{E}',g}(R) = \begin{cases} -\infty, & \text{if } f_G^{\mathcal{E}'}(R) + g(t) < \max_{R \in \mathcal{C}(I)} f_G^{\mathcal{E}'}(R) \\ f_G^{\mathcal{E}'}(R), & \text{otherwise.} \end{cases} \quad (5)$$

3.2 Equivalence relations and representatives

An encoder \mathcal{E} together with a function $g : \mathbb{N} \rightarrow \mathbb{N}$ define an equivalence relation $\sim_{\mathcal{E},g,t}$ on graphs in \mathcal{F}_t as follows.

Definition 9 (Equivalence relation $\sim_{\mathcal{E},g,t}$). Let \mathcal{E} be an encoder, let $g : \mathbb{N} \rightarrow \mathbb{N}$, and let $G_1, G_2 \in \mathcal{F}_t$. We say that $G_1 \sim_{\mathcal{E},g,t} G_2$ if and only if $\Lambda(G_1) = \Lambda(G_2) =: I$ and there exists an integer c , depending only on G_1 and G_2 , such that for every \mathcal{C} -encoding $R \in \mathcal{C}(I)$ it holds that

$$f_{G_1}^{\mathcal{E},g}(R) = f_{G_2}^{\mathcal{E},g}(R) + c. \quad (6)$$

Note that if there exists $R \in \mathcal{C}(I)$ such that $f_{G_1}^{\mathcal{E},g}(R) \neq \infty$, then the integer c satisfying Equation (6) is unique, otherwise every integer c satisfies Equation (6). We define the following function $\Delta_{\mathcal{E},g,t} : \mathcal{F}_t \times \mathcal{F}_t \rightarrow \mathbb{Z}$, which is called, following the terminology from Bodlaender *et al.* [7], the *transposition function* for the equivalence relation $\sim_{\mathcal{E},g,t}$.

$$\Delta_{\mathcal{E},g,t}(G_1, G_2) = \begin{cases} c, & \text{if } G_1 \sim_{\mathcal{E},g,t} G_2 \text{ and Eq. (6) holds for a unique integer } c; \\ 0, & \text{if } G_1 \sim_{\mathcal{E},g,t} G_2 \text{ and Eq. (6) holds for every integer; and} \\ \text{undefined} & \text{otherwise} \end{cases} \quad (7)$$

If we are dealing with a problem defined on a graph class \mathcal{G} , the protrusion replacement rule has to preserve the class \mathcal{G} , as otherwise we would obtain a *bikernel* instead of a

kernel. That is, we need to make sure that, when replacing a graph in $\mathcal{F}_t \cap \mathcal{G}$ with one of its representatives, we do not produce a graph that does not belong to \mathcal{G} anymore. To this end, we define an equivalence relation $\sim_{\mathcal{E},g,t,\mathcal{G}}$ on graphs in $\mathcal{F}_t \cap \mathcal{G}$, which refines the equivalence relation $\sim_{\mathcal{E},g,t}$ of Definition 9.

Definition 10 (Equivalence relation $\sim_{\mathcal{E},g,t,\mathcal{G}}$). *Let \mathcal{G} be a class of graphs and let $G_1, G_2 \in \mathcal{F}_t \cap \mathcal{G}$.*

- (i) $G_1 \sim_{\mathcal{G},t} G_2$ if and only if for any t -boundaried graph H , $G_1 \oplus H \in \mathcal{G}$ if and only if $G_2 \oplus H \in \mathcal{G}$.
- (ii) $G_1 \sim_{\mathcal{E},g,t,\mathcal{G}} G_2$ if and only if $G_1 \sim_{\mathcal{E},g,t} G_2$ and $G_1 \sim_{\mathcal{G},t} G_2$.

It is well-known by Büchi's theorem that regular languages are precisely those definable in Monadic Second Order logic (MSO logic). By Myhill-Nerode's theorem, it follows that if the membership in a graph class \mathcal{G} can be expressed in MSO logic, then the equivalence relation $\sim_{\mathcal{G},t}$ has a finite number of equivalence classes (see for instance [15, 16]). However, we do not have in general an explicit upper bound on the number of equivalence classes of $\sim_{\mathcal{G},t}$, henceforth denoted by $r_{\mathcal{G},t}$. Fortunately, in the context of our applications in Sections 4, 5, and 6, where \mathcal{G} will be a class of graphs that exclude some fixed graph as a (topological) minor², this will always be possible, and in this case it holds that $r_{\mathcal{G},t} \leq 2^{t \log t} \cdot h^t \cdot 2^{h^2}$.

For an encoder $\mathcal{E} = (\mathcal{C}, L_{\mathcal{C}})$, we let $s_{\mathcal{E}}(t) := \max_{I \subseteq \{1, \dots, t\}} |\mathcal{C}(I)|$, where $|\mathcal{C}(I)|$ denotes the number of \mathcal{C} -encodings in $\mathcal{C}(I)$. The following lemma gives an upper bound on the number of equivalence classes of $\sim_{\mathcal{E},g,t,\mathcal{G}}$, which depends also on $r_{\mathcal{G},t}$.

Lemma 1. *Let \mathcal{G} be a graph class whose membership can be expressed in MSO logic. For any encoder \mathcal{E} , any function $g : \mathbb{N} \rightarrow \mathbb{N}$, and any positive integer t , the equivalence relation $\sim_{\mathcal{E},g,t,\mathcal{G}}$ has finite index. More precisely, the number of equivalence classes of $\sim_{\mathcal{E},g,t,\mathcal{G}}$ is at most $r(\mathcal{E}, g, t, \mathcal{G}) := (g(t) + 2)^{s_{\mathcal{E}}(t)} \cdot 2^t \cdot r_{\mathcal{G},t}$.*

Proof: Let us first show that the equivalence relation $\sim_{\mathcal{E},g,t}$ has finite index. Indeed, let $I \subseteq \{1, \dots, t\}$. By definition, we have that for any graph $G \in \mathcal{F}_t$ with $\Lambda(G) = I$, the function $f_G^{\mathcal{E},g}$ can take at most $g(t) + 2$ distinct values ($g(t) + 1$ finite values and possibly the value $+\infty$). Therefore, it follows that the number of equivalence classes of $\sim_{\mathcal{E},g,t}$ containing all graphs G in \mathcal{F}_t with $\Lambda(G) = I$ is at most $(g(t) + 2)^{|\mathcal{C}(I)|}$. As the number of subsets of $\{1, \dots, t\}$ is 2^t , we deduce that the overall number of equivalence classes of $\sim_{\mathcal{E},g,t}$ is at most $(g(t) + 2)^{s_{\mathcal{E}}(t)} \cdot 2^t$. Finally, since the equivalence relation $\sim_{\mathcal{E},t,\mathcal{G}}$ is the Cartesian product of the equivalence relations $\sim_{\mathcal{E},g,t}$ and $\sim_{\mathcal{G},t}$, the result follows from the fact that \mathcal{G} can be expressed in MSO logic. \square

In order for an encoding \mathcal{E} and a function g to be useful for performing dynamic programming on graphs in \mathcal{F}_t that belong to a graph class \mathcal{G} , we introduce the following definition, which captures the natural fact that the tables of a dynamic programming algorithm should depend exclusively on the tables of the descendants in a rooted tree-decomposition. Before moving to the definition, we note that given a graph $G \in \mathcal{F}_t$ and a rooted tree-decomposition (T, \mathcal{X}) of G such that $\partial(G)$ is contained in the root-bag of (T, \mathcal{X}) , the labels of $\partial(G)$ can be propagated in a natural way to all bags of (T, \mathcal{X}) by introducing, removing, and shifting labels appropriately. Therefore, for any node x of T ,

² A particular case of the classes of graphs whose membership can be expressed in MSO logic. We would like to stress here that we rely on the expressibility of the *graph class* \mathcal{G} in MSO logic, whereas in previous work [7, 18, 22] what is used in the expressibility in CMSO logic of the *problems* defined on a graph class.

the graph G_x can be naturally seen as a graph in \mathcal{F}_t . (A brief discussion can be found in the proof of Lemma 4, and we refer to [7] for more details.)

Definition 11 (DP-friendly equivalence relation). *An equivalence relation $\sim_{\mathcal{E},g,t,\mathcal{G}}$ is DP-friendly if for any graph $G \in \mathcal{F}_t$ and any rooted tree-decomposition (T, \mathcal{X}) of G such that $\partial(G)$ is contained in the root-bag of (T, \mathcal{X}) , and for any descendant x of the root r of T , if G' is the graph obtained from G by replacing the graph $G_x \in \mathcal{F}_t$ with a graph $G'_x \in \mathcal{F}_t$ such that $G_x \sim_{\mathcal{E},g,t,\mathcal{G}} G'_x$, then G' satisfies the following conditions:*

- (i) $G \sim_{\mathcal{E},g,t,\mathcal{G}} G'$; and
- (ii) $\Delta_{\mathcal{E},g,t}(G, G') = \Delta_{\mathcal{E},g,t}(G_x, G'_x)$.

In Definition 11, as well as in the remainder of the article, when we *replace* the graph G_x with the graph G'_x , we do *not* remove from G any of the edges with both endvertices in the boundary of G_x . That is, $G' = (G - (V(G_x) - \partial(V(G_x)))) \oplus G'_x$.

Recall that for the protrusion replacement to be valid for a problem Π , the equivalence relation $\sim_{\mathcal{E},g,t,\mathcal{G}}$ needs to be a refinement of the canonical equivalence relation $\equiv_{\Pi,t}$ (note that this implies, in particular, that if $\sim_{\mathcal{E},g,t,\mathcal{G}}$ has finite index, then Π has FII). The next lemma states a sufficient condition for this property, and furthermore it gives the value of the transposition constant $\Delta_{\Pi,t}(G_1, G_2)$, which will be needed in order to update the parameter after the replacement.

Lemma 2. *Let Π be a vertex-certifiable problem. If \mathcal{E} is a Π -encoder and $\sim_{\mathcal{E},g,t,\mathcal{G}}$ is a DP-friendly equivalence relation, then for any two graphs $G_1, G_2 \in \mathcal{F}_t$ such that $G_1 \sim_{\mathcal{E},g,t,\mathcal{G}} G_2$, it holds that $G_1 \equiv_{\Pi,t} G_2$ and $\Delta_{\Pi,t}(G_1, G_2) = \Delta_{\mathcal{E},g,t}(G_1, G_2)$.*

Proof: Assume without loss of generality that Π is a minimization problem, and let $\mathcal{E} = (\mathcal{C}, L_{\mathcal{C}})$. We need to prove that for any t -boundaried graph H and any integer k , $(G_1 \oplus H, k) \in \Pi$ if and only if $(G_2 \oplus H, k + \Delta_{\mathcal{E},g,t}(G_1, G_2)) \in \Pi$. Suppose that $(G_1 \oplus H, k) \in \Pi$ (by symmetry the same arguments apply starting with G_2). This means that there exists $S_1 \subseteq V(G_1 \oplus H)$ with $|S_1| \leq k$ such that $(G_1 \oplus H, S_1) \in L_{\Pi}$. And since $G_1 \oplus H$ is a 0-boundaried graph and \mathcal{E} is a Π -encoder, we have that $(G_1 \oplus H, S_1, R_{\emptyset}) \in L_{\mathcal{C}}$, where $\mathcal{C}(\emptyset) = \{R_{\emptyset}\}$. This implies that

$$f_{G_1 \oplus H}^{\mathcal{E}}(R_{\emptyset}) \leq |S_1| \leq k. \quad (8)$$

As $\sim_{\mathcal{E},g,t,\mathcal{G}}$ is DP-friendly and $G_1 \sim_{\mathcal{E},g,t,\mathcal{G}} G_2$, it follows that $(G_1 \oplus H) \sim_{\mathcal{E},g,t,\mathcal{G}} (G_2 \oplus H)$ and that $\Delta_{\mathcal{E},g,t}(G_1 \oplus H, G_2 \oplus H) = \Delta_{\mathcal{E},g,t}(G_1, G_2)$. Since $G_2 \oplus H$ is also a 0-boundaried graph, the latter property and Equation (8) imply that

$$f_{G_2 \oplus H}^{\mathcal{E}}(R_{\emptyset}) = f_{G_1 \oplus H}^{\mathcal{E}}(R_{\emptyset}) + \Delta_{\mathcal{E},g,t}(G_1, G_2) \leq k + \Delta_{\mathcal{E},g,t}(G_1, G_2). \quad (9)$$

From Equation (9) it follows that there exists $S_2 \subseteq V(G_2 \oplus H)$ with $|S_2| \leq k + \Delta_{\mathcal{E},g,t}(G_1, G_2)$ such that $(G_2 \oplus H, S_2, R_{\emptyset}) \in L_{\mathcal{C}}$. Since $G_2 \oplus H$ is a 0-boundaried graph and \mathcal{E} is a Π -encoder, this implies that $(G_2 \oplus H, S_2) \in L_{\Pi}$, which in turn implies that $(G_2 \oplus H, k + \Delta_{\mathcal{E},g,t}(G_1, G_2)) \in \Pi$, as we wanted to prove. \square

The following definition will be important to guarantee that, when applying our protrusion replacement rule, the parameter of the problem under consideration does not increase.

Definition 12 (Progressive representatives of $\sim_{\mathcal{E},g,t,\mathcal{G}}$). *Let \mathfrak{C} be some equivalence class of $\sim_{\mathcal{E},g,t,\mathcal{G}}$ and let $G \in \mathfrak{C}$. We say that G is a progressive representative of \mathfrak{C} if for any graph $G' \in \mathfrak{C}$ it holds that $\Delta_{\mathcal{E},g,t}(G, G') \leq 0$.*

In the next lemma we provide an upper bound on the size of a smallest *progressive* representative of any equivalence class of $\sim_{\mathcal{E},g,t,\mathcal{G}}$.

Lemma 3. *Let \mathcal{G} be a graph class whose membership can be expressed in MSO logic. For any encoder \mathcal{E} , any function $g : \mathbb{N} \rightarrow \mathbb{N}$, and any $t \in \mathbb{N}$ such that $\sim_{\mathcal{E},g,t,\mathcal{G}}$ is DP-friendly, every equivalence class of $\sim_{\mathcal{E},g,t,\mathcal{G}}$ has a progressive representative of size at most $b(\mathcal{E},g,t,\mathcal{G}) := 2^{r(\mathcal{E},g,t,\mathcal{G})+1} \cdot t$, where $r(\mathcal{E},g,t,\mathcal{G})$ is the function defined in Lemma 1.*

Proof: Let \mathfrak{C} be an arbitrary equivalence class of $\sim_{\mathcal{E},g,t,\mathcal{G}}$, and we want to prove that there exists in \mathfrak{C} a progressive representative of the desired size. Let us first argue that \mathfrak{C} contains some progressive representative. We construct an (infinite) directed graph $D_{\mathfrak{C}}$ as follows. There is a vertex in $D_{\mathfrak{C}}$ for every graph in \mathfrak{C} , and for any two vertices $v_1, v_2 \in V(D_{\mathfrak{C}})$, corresponding to two graphs $G_1, G_2 \in \mathfrak{C}$ respectively, there is an arc from v_1 to v_2 if and only if $\Delta_{\mathcal{E},g,t}(G_1, G_2) > 0$. We want to prove that $D_{\mathfrak{C}}$ has a sink, that is, a vertex with no outgoing arc, which by construction is equivalent to the existence of a progressive representative in \mathfrak{C} . Indeed, let v be an arbitrary vertex of $D_{\mathfrak{C}}$, and grow greedily a directed path P starting from v . Because of the transitivity of the equivalence relation $\sim_{\mathcal{E},g,t,\mathcal{G}}$ and by construction of $D_{\mathfrak{C}}$, it follows that $D_{\mathfrak{C}}$ does not contain any finite cycle, so P cannot visit vertex v again. On the other hand, since the function $f_G^{\mathcal{E}}$ takes only positive values (except possibly for the value $-\infty$), it follows that there are no arbitrarily long directed paths in $D_{\mathfrak{C}}$ starting from any fixed vertex, so in particular the path P must be finite, and therefore the last vertex in P is necessarily a sink. (Note that for any two graphs $G_1, G_2 \in \mathfrak{C}$ such that their corresponding vertices $v_1, v_2 \in V(D_{\mathfrak{C}})$ are sinks, it holds by construction of $D_{\mathfrak{C}}$ that $\Delta_{\mathcal{E},g,t}(G_1, G_2) = 0$.)

Now let $G \in \mathcal{F}_t \cap \mathcal{G}$ be a progressive representative of \mathfrak{C} with minimum number of vertices. We claim that G has size at most $2^{r(\mathcal{E},g,t,\mathcal{G})+1} \cdot t$. (We would like to stress that at this stage we only need to care about the *existence* of such representative G , and not about how to *compute* it.) Indeed, let (T, \mathcal{X}) be a nice rooted tree-decomposition of G of width at most $t-1$ such that $\partial(G)$ is contained in the root-bag (such a nice tree-decomposition exists by [23]), and let r be the root of T .

We first claim that for any node x of T , the graph G_x is a progressive representative of its equivalence class with respect to $\sim_{\mathcal{E},g,t,\mathcal{G}}$, namely \mathfrak{A} . Indeed, assume that it is not the case, and let H be a progressive representative of \mathfrak{A} , which exists by the discussion in the first paragraph of the proof. Since H is progressive and G_x is not, $\Delta_{\mathcal{E},g,t}(H, G_x) < 0$. Let G_H be the graph obtained from G by replacing G_x with H . Since $\sim_{\mathcal{E},g,t,\mathcal{G}}$ is DP-friendly, it follows that $G \sim_{\mathcal{E},g,t,\mathcal{G}} G_H$ and that $\Delta_{\mathcal{E},g,t}(G_H, G) = \Delta_{\mathcal{E},g,t}(H, G_x) < 0$, contradicting the fact that G is a progressive representative of the equivalence class \mathfrak{C} .

We now claim that for any two nodes $x, y \in V(T)$ lying on a path from r to a leaf of T , it holds that $G_x \sim_{\mathcal{E},g,t,\mathcal{G}} G_y$. Indeed, assume for contradiction that there are two nodes $x, y \in V(T)$ lying on a path from r to a leaf of T such that $G_x \not\sim_{\mathcal{E},g,t,\mathcal{G}} G_y$. Let \mathfrak{A} be the equivalence class of G_x and G_y with respect to $\sim_{\mathcal{E},g,t,\mathcal{G}}$. By the previous claim, it follows that both G_x and G_y are progressive representatives of \mathfrak{A} , and therefore it holds that $\Delta_{\mathcal{E},g,t}(G_y, G_x) = 0$. Suppose without loss of generality that $G_y \subsetneq G_x$ (that is, G_y is a strict subgraph of G_x), and let G' be the graph obtained from G by replacing G_x with G_y . Again, since $\sim_{\mathcal{E},g,t,\mathcal{G}}$ is DP-friendly, it follows that $G \sim_{\mathcal{E},g,t,\mathcal{G}} G'$ and that $\Delta_{\mathcal{E},g,t}(G', G) = \Delta_{\mathcal{E},g,t}(G_y, G_x) = 0$. Therefore, G' is a progressive representative of \mathfrak{C} with $|V(G')| < |V(G)|$, contradicting the minimality of $|V(G)|$.

Finally, since for any two nodes $x, y \in V(T)$ lying on a path from r to a leaf of T we have that $G_x \sim_{\mathcal{E},g,t,\mathcal{G}} G_y$, it follows that the height of T is at most the number of equivalence classes of $\sim_{\mathcal{E},g,t,\mathcal{G}}$, which is at most $r(\mathcal{E},g,t,\mathcal{G})$ by Lemma 1. Since T is a binary tree, we have that $|V(T)| \leq 2^{r(\mathcal{E},g,t,\mathcal{G})+1} - 1$. Finally, since $|V(G)| \leq |V(T)| \cdot t$, it

follows that $|V(G)| \leq 2^{r(\mathcal{E},g,t,\mathcal{G})+1} \cdot t$, as we wanted to prove. \square

The next lemma states that if one is given an upper bound on the size of the progressive representatives of an equivalence relation defined on t -protrusions (that is, on graphs in \mathcal{F}_t)³, then a *small* progressive representative of a t -protrusion can be explicitly calculated in linear time. In other words, it provides a generic and constructive way to perform a dynamic programming procedure to replace protrusions, without needing to deal with the particularities of each encoder in order to compute the tables. Its proof uses some ideas taken from [7, 18].

Lemma 4. *Let \mathcal{G} be a graph class, let \mathcal{E} be an encoder, let $g : \mathbb{N} \rightarrow \mathbb{N}$, and let $t \in \mathbb{N}$ such that $\sim_{\mathcal{E},g,t,\mathcal{G}}$ is DP-friendly. Assume that we are given an upper bound b on the size of a smallest progressive representative of any equivalence class of $\sim_{\mathcal{E},g,t,\mathcal{G}}$. Then, given an n -vertex t -protrusion G , we can output in time $O(n)$ a t -protrusion H of size at most b such that $G \sim_{\mathcal{E},g,t,\mathcal{G}} H$ and the corresponding transposition constant $\Delta_{\mathcal{E},g,t}(H, G)$ with $\Delta_{\mathcal{E},g,t}(H, G) \leq 0$, where the hidden constant in the “ O ” notation depends only on $\mathcal{E}, g, b, \mathcal{G}$, and t .*

Proof: Let $\mathcal{E} = (\mathcal{C}, L_{\mathcal{C}})$ be the given encoder. We start by generating a repository \mathfrak{R} containing all the graphs in \mathcal{F}_t with at most $b + 1$ vertices. Such a set of graphs, as well as a rooted nice tree-decomposition of width at most $t - 1$ of each of them, can be clearly generated in time depending only on b and t . By assumption, the size of a smallest progressive representative of any equivalence class of $\sim_{\mathcal{E},g,t,\mathcal{G}}$ is at most b , so \mathfrak{R} contains a progressive representative of any equivalence class of $\sim_{\mathcal{E},g,t,\mathcal{G}}$ with at most b vertices. We now partition the graphs in \mathfrak{R} into equivalence classes of $\sim_{\mathcal{E},g,t,\mathcal{G}}$ as follows. For each graph $H \in \mathfrak{R}$ and each \mathcal{C} -encoding $R \in \mathcal{C}(\Lambda(G))$, as $L_{\mathcal{C}}$ is a computable language, we can compute the value $f_G^{\mathcal{E},g}(R)$ in time depending only on \mathcal{E}, g, t , and b . Therefore, for any two graphs $H_1, H_2 \in \mathfrak{R}$, we can decide in time depending only on \mathcal{E}, g, t, b , and \mathcal{G} whether $H_1 \sim_{\mathcal{E},g,t,\mathcal{G}} H_2$, and if this is the case, we can compute the transposition constant $\Delta_{\mathcal{E},g,t,\mathcal{G}}(H_1, H_2)$ within the same running time.

Given a t -protrusion G on n vertices with boundary $\partial(G)$, we first compute a rooted nice tree-decomposition (T, \mathcal{X}) of G such that $\partial(G)$ is contained in the root bag in time $f(t) \cdot n$, by using the linear-time algorithm of Bodlaender [4]. Such a t -protrusion G equipped with a tree-decomposition can be naturally seen as a graph in \mathcal{F}_t by assigning distinct labels from $\{1, \dots, t\}$ to the vertices in the root-bag. These labels from $\{1, \dots, t\}$ can be transferred to the vertices in all the bags of (T, \mathcal{X}) by performing a standard shifting procedure when a vertex is introduced or removed from the nice tree-decomposition (see [7] for more details). Therefore, each node $x \in V(T)$ defines in a natural way a graph $G_x \subseteq G$ in \mathcal{F}_t with its associated rooted nice tree-decomposition. Let us now proceed to the description of the replacement algorithm.

We process the bags of (T, \mathcal{X}) in a bottom-up way until we encounter the first node x in $V(T)$ such that $|V(G_x)| = b + 1$. (Note that as (T, \mathcal{X}) is a nice tree-decomposition, when processing the bags in a bottom-up way, at most one new vertex is introduced at every step.) Let \mathfrak{C} be the equivalence class of G_x according to $\sim_{\mathcal{E},g,t,\mathcal{G}}$. As $|V(G_x)| = b + 1$, the graph G_x is contained in the repository \mathfrak{R} , so in constant time we can find in \mathfrak{R} a progressive representative F of \mathfrak{C} with at most b vertices and the corresponding transposition constant $\Delta_{\mathcal{E},g,t}(F, G_x) \leq 0$, where the inequality holds because F is a progressive representative. Let G' be the graph obtained from G by replacing G_x with

³ Note that we slightly abuse notation when identifying t -protrusions and graphs in \mathcal{F}_t , as protrusions are defined as subsets of vertices of a graph. Nevertheless, this will not cause any confusion.

F , so we have that $|V(G')| < |V(G)|$. (Note that this replacement operation directly yields a rooted nice tree-decomposition of width at most $t - 1$ of G' .) Since $\sim_{\mathcal{E},g,t,\mathcal{G}}$ is DP-friendly, it follows that $G \sim_{\mathcal{E},g,t,\mathcal{G}} G'$ and that $\Delta_{\mathcal{E},g,t}(G', G) = \Delta_{\mathcal{E},g,t}(F, G_x) \leq 0$.

We recursively apply this replacement procedure on the resulting graph until we eventually obtain a t -protrusion H with at most b vertices such that $G \sim_{\mathcal{E},g,t,\mathcal{G}} H$. The corresponding transposition constant $\Delta_{\mathcal{E},g,t}(H, G)$ can be easily computed by summing up all the transposition constants given by each of the performed replacements. Since each of these replacements introduces a progressive representative, we have that $\Delta_{\mathcal{E},g,t}(H, G) \leq 0$. As we can assume that the total number of nodes in a nice tree-decomposition of G is $O(k)$ [23, Lemma 13.1.2], the overall running time of the algorithm is $O(n)$, where the constant hidden in the “ O ” notation depends indeed exclusively on $\mathcal{E}, g, b, \mathcal{G}$, and t . \square

3.3 Explicit protrusion replacer

We are now ready to piece everything together and state our main technical result, which can be interpreted as a generic *constructive* way of performing protrusion replacement with *explicit* size bounds. For our algorithms to be fully constructive, we restrict \mathcal{G} to be the class of graphs that exclude some fixed graph H as a (topological) minor.

Theorem 1. *Let H be a fixed graph and let \mathcal{G} be the class of graphs that exclude H as a (topological) minor. Let Π be a vertex-certifiable parameterized graph problem defined on \mathcal{G} , and suppose that we are given a Π -encoder \mathcal{E} , a function $g : \mathbb{N} \rightarrow \mathbb{N}$, and an integer $t \in \mathbb{N}$ such that $\sim_{\mathcal{E},g,t,\mathcal{G}}$ is DP-friendly. Then, given an input graph (G, k) and a t -protrusion Y in G , we can compute in time $O(|Y|)$ an equivalent instance $((G - (Y - \partial(Y))) \oplus Y', k')$, where $k' \leq k$ and Y' is a t -protrusion with $|Y'| \leq b(\mathcal{E}, g, t, \mathcal{G})$, where $b(\mathcal{E}, g, t, \mathcal{G})$ is the function defined in Lemma 3.*

Proof: By Lemma 1, the number of equivalence classes of the equivalence relation $\sim_{\mathcal{E},g,t,\mathcal{G}}$ is finite, and by Lemma 3 the size of a smallest progressive representative of any equivalence class of $\sim_{\mathcal{E},g,t,\mathcal{G}}$ is at most $b(\mathcal{E}, g, t, \mathcal{G})$. Therefore, we can apply Lemma 4 and deduce that, in time $O(|Y|)$, we can find a t -protrusion Y' of size at most $b(\mathcal{E}, g, t, \mathcal{G})$ such that $Y \sim_{\mathcal{E},g,t,\mathcal{G}} Y'$, and the corresponding transposition constant $\Delta_{\mathcal{E},g,t}(Y', Y)$ with $\Delta_{\mathcal{E},g,t}(Y', Y) \leq 0$. Since \mathcal{E} is a Π -encoder and $\sim_{\mathcal{E},g,t,\mathcal{G}}$ is DP-friendly, it follows from Lemma 2 that $Y \equiv_{\Pi,t} Y'$ and that $\Delta_{\Pi,t}(Y', Y) = \Delta_{\mathcal{E},g,t}(Y', Y) \leq 0$. Therefore, if we set $k' := k + \Delta_{\Pi,t}(Y', Y)$, it follows that (G, k) and $((G - (Y - \partial(Y))) \oplus Y', k')$ are indeed equivalent instances of Π with $k' \leq k$ and $|Y'| \leq b(\mathcal{E}, g, t, \mathcal{G})$. \square

The general recipe to use our framework on a parameterized problem Π defined on a class of graphs \mathcal{G} is as follows: one has just to define the tables to solve Π via dynamic programming on graphs of bounded treewidth (that is, the encoder \mathcal{E} and the function g), check that \mathcal{E} is a Π -encoder and that $\sim_{\mathcal{E},g,t,\mathcal{G}}$ is DP-friendly, and then Theorem 1 provides a linear-time algorithm that replaces large protrusions with graphs whose size is bounded by an explicit constant, and that updates the parameter of Π accordingly. This protrusion replacer can then be used, for instance, whenever one is able to find a linear protrusion decomposition of the input graphs of Π on some sparse graph class \mathcal{G} . In particular, Theorem 1 yields the following corollary.

Corollary 1. *Let H be a fixed graph, and let \mathcal{G} be the class of graphs that exclude H as a (topological) minor. Let Π be a vertex-certifiable parameterized graph problem on \mathcal{G} , and suppose that we are given a Π -encoder \mathcal{E} , a function $g : \mathbb{N} \rightarrow \mathbb{N}$, and an integer $t \in \mathbb{N}$*

such that $\sim_{\mathcal{E},g,t,\mathcal{G}}$ is DP-friendly. Then, given an instance (G, k) of Π together with an $(\alpha \cdot k, t)$ -protrusion decomposition of G , we can construct a linear kernel for Π of size at most $(1 + b(\mathcal{E}, g, t, \mathcal{G})) \cdot \alpha \cdot k$, where $b(\mathcal{E}, g, t, \mathcal{G})$ is the function defined in Lemma 3.

Proof: For $1 \leq i \leq \ell$, we apply the polynomial-time algorithm given by Theorem 1 to replace each t -protrusion Y_i with a graph Y'_i of size at most $b(\mathcal{E}, g, t, \mathcal{G})$, and to update the parameter accordingly. In this way we obtain an equivalent instance (G', k') such that $G' \in \mathcal{G}$, $k' \leq k$, and $|V(G')| \leq |Y_0| + \ell \cdot b(\mathcal{E}, g, t, \mathcal{G}) \leq (1 + b(\mathcal{E}, g, t, \mathcal{G}))\alpha \cdot k$. \square

Notice that once we fix the problem Π and the class of graphs \mathcal{G} where Corollary 1 is applied, a kernel of size $c \cdot k$ can be derived with a concrete upper bound for the value of c . Notice that such a bound depends on the problem Π and the excluded (topological) minor H . In general, the bound can be quite big as it depends on the bound of Lemma 3, and this, in turn, depends on the bound of Lemma 1. However, as we see in the next section, more moderate estimations can be extracted for particular families of parameterized problems.

Before demonstrating the applicability of our framework by providing linear kernels for several families of problems on graphs excluding a fixed graph as a (topological) minor, we need another ingredient. Namely, the following result will be fundamental in order to find linear protrusion decompositions when a treewidth-modulator X of the input graph G is given, with $|X| = O(k)$. It is a consequence of [22, Lemma 3, Proposition 1, and Theorem 1] and, loosely speaking, the algorithm consists in marking the bags of a tree-decomposition of $G - X$ according to the number of neighbors in the set X . When the graph G is restricted to exclude a fixed graph H as a topological minor, it can be proved that the obtained protrusion decomposition is linear. All the details can be found in the full version of [22].

Theorem 2 (Kim et al. [22]). *Let c, t be two positive integers, let H be an h -vertex graph, let G be an n -vertex H -topological-minor-free graph, and let k be a positive integer (typically corresponding to the parameter of a parameterized problem). If we are given a set $X \subseteq V(G)$ with $|X| \leq c \cdot k$ such that $\text{tw}(G - X) \leq t$, then we can compute in time $O(n)$ an $((\alpha_H \cdot t \cdot c) \cdot k, 2t + h)$ -protrusion decomposition of G , where α_H is a constant depending only on H , which is upper-bounded by $40h^2 2^{5h \log h}$.*

As mentioned in Subsection 3.2, if \mathcal{G} is a graph class whose membership can be expressed in MSO logic, then $\sim_{\mathcal{G},t}$ has a finite number of equivalence classes, namely $r_{\mathcal{G},t}$. In our applications, we will be only concerned with families of graphs \mathcal{G} that exclude some fixed h -vertex graph H as a (topological) minor. In this case, using standard dynamic programming techniques, it can be shown that $r_{\mathcal{G},t} \leq 2^{t \log t} \cdot h^t \cdot 2^{h^2}$. The details can be found in the encoder described in Subsection 6.1 for the \mathcal{F} -DELETION problem.

4 An explicit linear kernel for r -Dominating Set

Let $r \geq 1$ be a fixed integer. We define the r -DOMINATING SET problem as follows.

r -DOMINATING SET

Instance: A graph $G = (V, E)$ and a non-negative integer k .

Parameter: The integer k .

Question: Does G have a set $S \subseteq V$ with $|S| \leq k$ and such that every vertex in $V \setminus S$ is within distance at most r from some vertex in S ?

For $r = 1$, the r -DOMINATING SET problem corresponds to DOMINATING SET. Our encoder for r -DOMINATING SET is strongly inspired by the work of Demaine *et al.* [13], and it generalizes to one given for DOMINATING SET in the running example of Section 3. The encoder for r -DOMINATING SET, which we call $\mathcal{E}_{r\text{DS}} = (\mathcal{C}_{r\text{DS}}, L_{\mathcal{C}_{r\text{DS}}})$, is described in Subsection 4.1, and we show how to construct the linear kernel in Subsection 4.2.

4.1 Description of the encoder

Let G be a boundaried graph with boundary $\partial(G)$ and let $I = \Lambda(G)$. The function $\mathcal{C}_{r\text{DS}}$ maps I to a set $\mathcal{C}_{r\text{DS}}(I)$ of $\mathcal{C}_{r\text{DS}}$ -encodings. Each $R \in \mathcal{C}_{r\text{DS}}(I)$ maps I to an $|I|$ -tuple in $\{0, \uparrow, \downarrow, 1, \dots, \uparrow, \downarrow, r\}^{|I|}$, and thus the coordinates of the tuple are in one-to-one correspondence with the vertices of $\partial(G)$. For a vertex $v \in \partial(G)$ we denote by $R(v)$ its coordinate in the $|I|$ -tuple. For a subset S of vertices of G , we say that (G, S, R) belongs to the language $L_{\mathcal{C}_{r\text{DS}}}$ (or that S is a *partial r -dominating set satisfying R*) if :

- for every vertex $v \in V(G) \setminus \partial(G)$, either $d_G(v, S) \leq r$ or there exists $w \in \partial(G)$ such that $R(w) = \uparrow j$ and $d_G(v, w) + j \leq r$; and
- for every vertex $v \in \partial(G)$: $R(v) = 0$ implies that $v \in S$, and if $R(v) = \downarrow i$ for $1 \leq i \leq r$, then there exists either $w \in S$ such that $d_G(v, w) \leq i$ or $w \in \partial(G)$ such that $R(w) = \uparrow j$ and $d_G(v, w) + j \leq i$.

Observe that if S is a partial r -dominating set satisfying R , then $S \cap \partial(G)$ contains the set of vertices $\{v \in \partial(G) \mid R(v) = 0\}$, but it may also contain other vertices of $\partial(G)$. As the optimization version of r -DOMINATING SET is a minimization problem, by Equation (1) the function $f_G^{\mathcal{C}_{r\text{DS}}}(R)$ associates with a $\mathcal{C}_{r\text{DS}}$ -encoding R the minimum size of a partial r -dominating set S satisfying R . By definition of $\mathcal{E}_{r\text{DS}}$, it is clear that

$$s_{\mathcal{E}_{r\text{DS}}}(t) \leq (2r + 1)^t. \quad (10)$$

Lemma 5. *The encoder $\mathcal{E}_{r\text{DS}}$ is a $r\text{DS}$ -encoder. Furthermore, if \mathcal{G} is an arbitrary class of graphs and $g(t) = t$, then the equivalence relation $\sim_{\mathcal{E}_{r\text{DS}}, g, t, \mathcal{G}}$ is DP-friendly.*

Before providing the proof of Lemma 5, we will first state a general fact, which will be useful in order to prove that an encoder is DP-friendly.

Fact 1 *To verify that an equivalence relation $\sim_{\mathcal{E}, g, t, \mathcal{G}}$ satisfies Definition 11, property (i) can be replaced with $G \sim_{\mathcal{E}, g, t} G'$. That is, if $G \sim_{\mathcal{E}, g, t} G'$, then $G \sim_{\mathcal{E}, g, t, \mathcal{G}} G'$ as well.*

Proof: Assume that $G \sim_{\mathcal{E}, g, t} G'$, and we want to deduce that $G \sim_{\mathcal{E}, g, t, \mathcal{G}} G'$, that is, we just have to prove that $G \sim_{\mathcal{G}, t} G'$. Let H be a t -boundaried graph, and we need to prove that $G \oplus H \in \mathcal{G}$ if and only if $G' \oplus H \in \mathcal{G}$. Let G^- such that $G = G_x \oplus G^-$, and note that $G' = G'_x \oplus G^-$. We have that $G \oplus H = (G_x \oplus G^-) \oplus H = G_x \oplus (G^- \oplus H)$, and similarly we have that $G'_x \oplus (G^- \oplus H) = (G'_x \oplus G^-) \oplus H = G' \oplus H$. Since $G_x \sim_{\mathcal{G}, t} G'_x$, it follows that $G \oplus H = G_x \oplus (G^- \oplus H) \in \mathcal{G}$ if and only if $G'_x \oplus (G^- \oplus H) = G' \oplus H \in \mathcal{G}$. \square

We will use the shortcut $r\text{DS}$ for r -DOMINATING SET.

Proof of Lemma 5: Let us first prove that $\mathcal{E}_{r\text{DS}} = (\mathcal{C}_{r\text{DS}}, L_{\mathcal{C}_{r\text{DS}}})$ is a $r\text{DS}$ -encoder. Note that there is a unique 0-tuple $R_\emptyset \in \mathcal{C}_{r\text{DS}}(\emptyset)$, and by definition of $L_{\mathcal{C}_{r\text{DS}}}$, $(G, S, R_\emptyset) \in L_{\mathcal{C}_{r\text{DS}}}$ if and only if S is an r -dominating set of G . Let us now prove that the equivalence relation $\sim_{\mathcal{E}_{r\text{DS}}, g, t, \mathcal{G}}$ is DP-friendly for $g(t) = t$.

Let G be a t -boundaried graph with boundary A , and consider a tree-decomposition of G of width at most $t - 1$ rooted at A . Let B be any bag of the tree-decomposition.

Each such bag B defines a subgraph G_B of G (recall that G_B can be viewed as a t -boundaried graph with boundary B). We define H to be the t -boundaried graph induced by $V(G) \setminus (V(G_B) \setminus B)$, and with boundary B (that is, we forget boundary A) labeled as in G_B . Let G'_B be a t -boundaried graph such that $G_B \sim_{\mathcal{E}_{r\text{DS},g,t,\mathcal{G}}} G'_B$. Let $G' := H \oplus G'_B$ with boundary A . See Fig. 2 for an illustration.

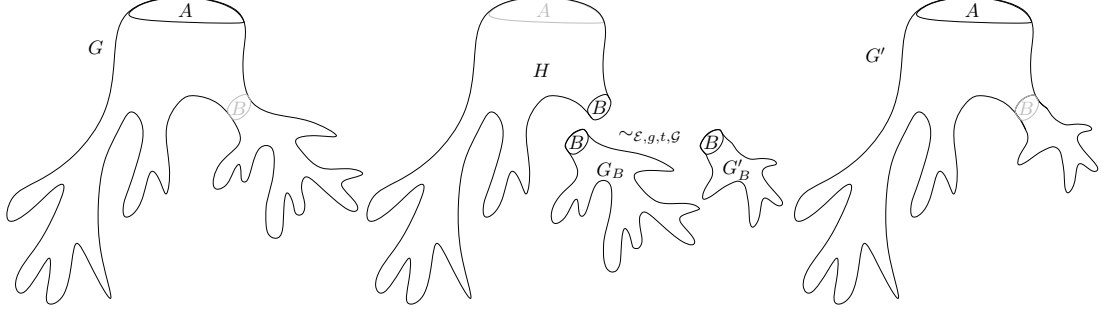


Fig. 2. Graphs G and G' in the proof of Lemma 5.

We claim that the encoder $\mathcal{E}_{r\text{DS}}$ is g -confined for $g(t) = t$. Indeed, consider an arbitrary encoding $R_A \in \mathcal{C}_{r\text{DS}}(\Lambda(G))$ and the encoding R_0 satisfying $R_0(v) = 0$ for every $v \in A$. Let $S_0 \subseteq V(G)$ be a minimum-sized partial r -dominating set satisfying R_0 , i.e., such that $(G, S_0, R_0) \in L_{\mathcal{C}_{r\text{DS}}}$. Observe that S_0 also satisfies R_A , i.e., $(G, S_0, R_A) \in L_{\mathcal{C}_{r\text{DS}}}$. It then follows that $f_G^{\mathcal{E}_{r\text{DS}}}(R_0) = \max_{R_A} f_G^{\mathcal{E}_{r\text{DS}}}(R_A)$. Moreover, let $S \subseteq V(G)$ be a minimum-sized partial r -dominating set satisfying R_A , i.e., such that $(G, S, R_A) \in L_{\mathcal{C}_{r\text{DS}}}$. Then R_0 is also satisfied by $S \cup A$. It follows that $f_G^{\mathcal{E}_{r\text{DS}}}(R_0) - \min_{R_A} f_G^{\mathcal{E}_{r\text{DS}}}(R_A) \leq |A| \leq t$, proving that the encoder is indeed g -confined.

We want to show that $G \sim_{\mathcal{E}_{r\text{DS},g,t,\mathcal{G}}} G'$ and that $\Delta_{\mathcal{E}_{r\text{DS},g,t}}(G, G') = \Delta_{\mathcal{E}_{r\text{DS},g,t}}(G_B, G'_B)$. According to Fact 1, we can consider the relation $\sim_{\mathcal{E}_{r\text{DS},g,t}}$ (that is, we do not need to consider the refinement with respect to the class of graphs \mathcal{G}), and due to the g -confinement it holds that $f_G^{\mathcal{E}_{r\text{DS},g}} = f_G^{\mathcal{E}_{r\text{DS}}}$ for $g(t) = t$. Hence it suffices to prove that $f_G^{\mathcal{E}_{r\text{DS}}}(R_A) = f_{G'}^{\mathcal{E}_{r\text{DS}}}(R_A) + \Delta_{\mathcal{E}_{r\text{DS},g,t}}(G_B, G'_B)$ for all $R_A \in \mathcal{C}_{r\text{DS}}(\Lambda(G))$.

Let $R_A \in \mathcal{C}_{r\text{DS}}(\Lambda(G))$ be a $\mathcal{C}_{r\text{DS}}$ -encoding defined on A . First assume that $f_G^{\mathcal{E}_{r\text{DS}}}(R_A) \neq +\infty$, that is, $R_A \in \mathcal{C}_{r\text{DS},G}^*$. Let $S = D \cup D_H$ be a partial r -dominating set of size $f_G^{\mathcal{E}_{r\text{DS}}}(R_A)$ of G satisfying R_A , with $D \subseteq V(G_B)$ and $D_H \subseteq V(H) \setminus B$. We use S to construct a $\mathcal{C}_{r\text{DS}}$ -encoding $R_B \in \mathcal{C}_{r\text{DS}}(\Lambda(G_B))$ defined on B , satisfied by D as follows. Let $v \in B$:

- if $v \in S$, then $R_B(v) = 0$;
- otherwise, if there is either a shortest path from v to S of length i or a path from v to any $a \in A$ such that $R_A(a) = \uparrow j$ of length $i - j$, in both cases with its first edge in G_B , then $R_B(v) = \downarrow i$;
- otherwise, $R_B(v) = \uparrow i$ where $i = d_G(v, S)$ or $i = d_G(v, a) + j$ such that $R_A(a) = \uparrow j$ (the first edge of any shortest path from v to S is not in G_B).

See Fig. 3(a) for an illustration of the construction of the $\mathcal{C}_{r\text{DS}}$ -encoding $R_B \in \mathcal{C}_{r\text{DS}}(\Lambda(G_B))$ described above.

Observe that by construction of R_B , $|D| \geq f_{G_B}^{\mathcal{E}_{r\text{DS}}}(R)$. Let D' be a subset of vertices of G'_B of minimum size such that $(G'_B, D', R_B) \in L_{\mathcal{C}_{r\text{DS}}}$, that is, $|D'| = f_{G'_B}^{\mathcal{E}_{r\text{DS}}}(R_B)$. As

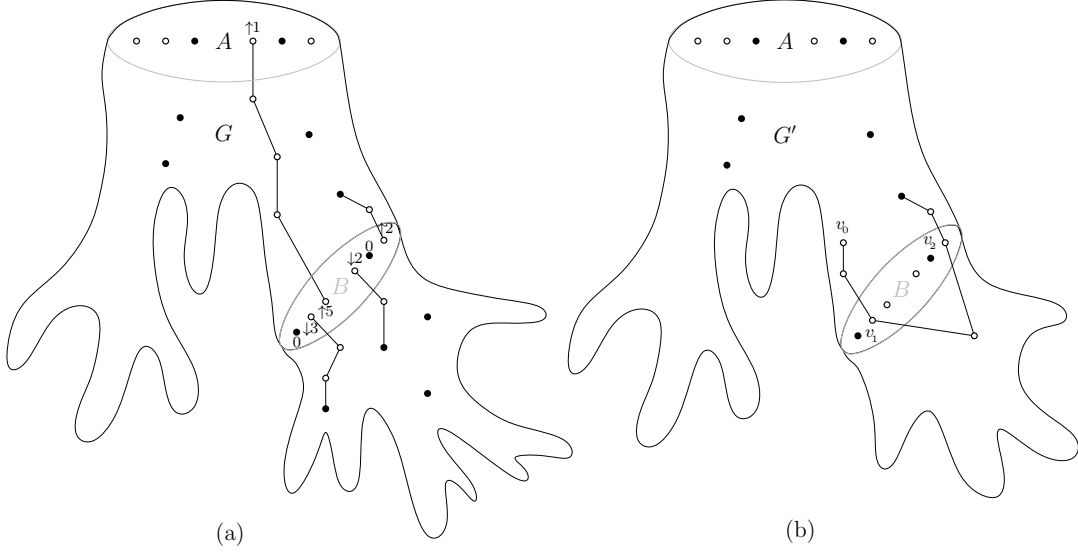


Fig. 3. Illustration of the proof of Lemma 5. Black vertices belong to the solution: (a) construction of the \mathcal{C}_{rDS} -encoding $R_B \in \mathcal{C}_{rDS}(\Lambda(G_B))$; and (b) construction of the corresponding paths.

$G_B \sim_{\mathcal{E}_{rDS}, g, t} G'_B$, we have $|D'| = f_{G_B}^{\mathcal{E}_{rDS}}(R_B) + \Delta_{\mathcal{E}_{rDS}, g, t}(G_B, G'_B)$ and therefore $|D' \cup D_H| = f_{G_B}^{\mathcal{E}_{rDS}}(R_A) + \Delta_{\mathcal{E}_{rDS}, g, t}(G_B, G'_B) + |D_H| \leq f_{G_B}^{\mathcal{E}_{rDS}}(R_A) + \Delta_{\mathcal{E}_{rDS}, g, t}(G_B, G'_B)$.

Let us now prove that $S' = D' \cup D_H$ is a partial r -dominating set of G' satisfying R_A . According to the definition of \mathcal{E}_{rDS} , we distinguish vertices in $V(G') \setminus A$ and in A .

We start with vertices not in A . For any vertex $v \in V(G') \setminus (A \cup S')$, we consider the following iterative process that builds a path of length at most r from v to S' or a path of length at most $r - i$ from v to $a \in A$ such that $R_A(a) = \uparrow i$. At step $j \geq 0$, we identify a vertex $v_j \in B$. We initially set $v_0 = v$. If $v_0 \in V(G'_B)$, we can assume that $d_{G'_B}(v_0, D') > r$, as otherwise we are done. As D' satisfies R_B , this implies that B contains a vertex v_1 such that $R_B(v_1) = \uparrow i_1$ and $d_{G'_B}(v_0, v_1) + i_1 \leq r$. Similarly, if $v_0 \in V(H) \setminus B$, we can assume that $d_H(v_0, D_H) > r$ and $d_H(v_0, a) > r - i$ for any $a \in A$ such that $R_A(a) = \uparrow i$, as otherwise we are done. As $S = D \cup D_H$ is a partial r -dominating set of G satisfying R_A , any shortest path P (of length at most r) between v_0 and S and any path (of length $r - i$) between v_0 and $a \in A$ such that $R_A(a) = \uparrow i$, contains a vertex of B incident to an edge of G_B . Let v_1 be the first such vertex of P . By definition of R_B , we have that $R_B(v_1) = \downarrow i_1$ with $d_H(v_0, v_1) + i_1 \leq r$. Let us now consider v_j with $j \geq 1$, and denote by l_j the length of the path we discovered from v_0 to v_j . We need to prove that $l_j + i_j \leq r$ (or $l_j + i_j \leq r - i$ in the other case) is an invariant of the process. As we argued, it is true for $j = 1$, so assume it holds at step j . We consider two cases:

1. $R_B(v_j) = \downarrow i_j$: We can assume that $d_{G'_B}(v_j, D') > i_j$, otherwise we are done as by construction it holds that $l_j + i_j \leq r$ (or $l_j + i_j \leq r - i$ in the other case). So as D' is a partial r -dominating set satisfying R_B , there exists a vertex $v_{j+1} \in B$ such that $R_B(v_{j+1}) = \uparrow i_{j+1}$ and $d_{G'_B}(v_j, v_{j+1}) + i_{j+1} \leq i_j$. As $l_{j+1} = l_j + d_{G'_B}(v_j, v_{j+1})$, it follows that $l_{j+1} + i_{j+1} \leq r$ (or $l_{j+1} + i_{j+1} \leq r - i$ in the other case). See Fig. 3(b) for an illustration of this case.
2. $R_B(v_j) = \uparrow i_j$: We can assume that $d_H(v_j, D_H) > i_j$ and $d_H(v_j, a) > i_j - i$ for any $a \in A$ such that $R_A(a) = \uparrow i$, otherwise we are done as by construction it holds that $l_j + i_j \leq r$ (or $l_j + i_j \leq r - i$ in the other case). As by definition of the encoding

R_B , $d_G(v_j, S) = i_j$, any shortest path P between v_j and S (or $a \in A$) uses a vertex of B incident to an edge of G_B . Let v_{j+1} be the first such vertex of P . Then $R_B(v_{j+1}) = \downarrow i_{j+1}$ with $d_H(v_j, v_{j+1}) + i_{j+1} \leq i_j$. As $l_{j+1} = l_j + d_H(v_j, v_{j+1})$, it follows that $l_{j+1} + i_{j+1} \leq r$ (or $l_j + i_j \leq r - i$ in the other case).

Observe that the process ends, since the parameter $r - (l_j + i_j)$ is strictly decreasing.

We now consider vertices of A . Note that as we consider a tree-decomposition and $\partial(G_B) = \partial(G'_B)$, it holds that $\partial(G) = \partial(G')$ as well. In particular, any vertex $v \in A$ is also in H . If $R_A(v) = 0$ since $S = D \cup D_H$ satisfies A , $v \in D_H$ and hence, $v \in S' = D' \cup D_H$. If $R_A(v) = \downarrow i$, the iterative process above built a path from v to S' of length at most r , or from v to $a \in A$ with $R_A(a) = \uparrow j$ of length at most $r - i - j$.

It follows that $S' = D' \cup D_H$ is a partial r -dominating set of size at most $f_G^{\mathcal{E}_{rDS}}(R_A) + \Delta_{\mathcal{E}_{rDS}, g, t}(G_B, G'_B)$ satisfying R_A , as we wanted to prove.

Finally, assume that $f_G^{\mathcal{E}_{rDS}}(R_A) = +\infty$. Then it holds that $f_{G'}^{\mathcal{E}_{rDS}}(R_A) = +\infty$ as well. Indeed, suppose that $f_{G'}^{\mathcal{E}_{rDS}}(R_A)$ is finite. Then, given a partial r -dominating set of G' satisfying R_A , by the argument above we could construct a partial r -dominating set of G satisfying R_A , contradicting that $f_G^{\mathcal{E}_{rDS}}(R_A) = +\infty$.

Therefore, we can conclude that $G \sim_{\mathcal{E}_{rDS}, g, t} G'$, and hence the equivalence relation $\sim_{\mathcal{E}_{rDS}, g, t, \mathcal{G}}$ is DP-friendly for $g(t) = t$. \square

4.2 Construction of the kernel

We proceed to construct a linear kernel for r -DOMINATING SET when the input graph excludes a fixed apex graph H as a minor. Toward this end, the following theorem will play an important role. It follows mainly from the results of Fomin *et al.* [18], but also uses the explicit combinatorial bound of Kawarabayashi and Kobayashi [20] on the relation between the treewidth and the largest grid minor on H -minor-free graphs, and the algorithmic results of Kawarabayashi and Reed [21] in order to obtain the claimed set X .

Theorem 3 (Fomin *et al.* [18]). *Let $r \geq 1$ be an integer, let H be an h -vertex apex graph, and let rDS_H be the restriction of the r -DOMINATING SET problem to input graphs which exclude H as a minor. If $(G, k) \in rDS_H$, then there exists a set $X \subseteq V(G)$ such that $|X| = r \cdot 2^{O(h \log h)} \cdot k$ and $\text{tw}(G - X) = r \cdot 2^{O(h \log h)}$. Moreover, given an instance (G, k) of rDS_H with $|V(G)| = n$, there is an algorithm running in time $O(n^3)$ that either finds such a set X or correctly reports that (G, k) is a NO-instance.*

We are now ready to present the linear kernel for r -DOMINATING SET.

Theorem 4. *Let $r \geq 1$ be an integer, let H be an h -vertex apex graph, and let rDS_H be the restriction of the r -DOMINATING SET problem to input graphs which exclude H as a minor. Then rDS_H admits a constructive linear kernel of size at most $f(r, h) \cdot k$, where f is an explicit function depending only on r and h , defined in Equation (11) below.*

Proof: Given an instance (G, k) of rDS_H , we run the cubic algorithm given by Theorem 3 to either conclude that (G, k) is a NO-instance or to find a set $X \subseteq V(G)$ such that $|X| = r \cdot 2^{O(h \log h)} \cdot k$ and $\text{tw}(G - X) = r \cdot 2^{O(h \log h)}$. In the latter case, we use the set X as input to the algorithm given by Theorem 2, which outputs in linear time a $(r^2 \cdot 2^{O(h \log h)} \cdot k, r \cdot 2^{O(h \log h)})$ -protrusion decomposition of G . We now consider the encoder $\mathcal{E}_{rDS} = (\mathcal{C}_{rDS}, L_{\mathcal{C}_{rDS}})$ defined in Subsection 4.1. By Lemma 5, \mathcal{E}_{rDS} is an rDS -encoder and $\sim_{\mathcal{E}_{rDS}, g, t, \mathcal{G}}$ is DP-friendly, where \mathcal{G} is the class of H -minor-free graphs and

$g(t) = t$. By Equation (10) in Subsection 4.1, we have that $s_{\mathcal{E}_{r\text{DS}}}(t) \leq (2r+1)^t$. Therefore, we are in position to apply Corollary 1 and obtain a linear kernel for $r\text{DS}_H$ of size at most

$$r^2 \cdot 2^{O(h \log h)} \cdot b\left(\mathcal{E}_{r\text{DS}}, g, r \cdot 2^{O(h \log h)}, \mathcal{G}\right) \cdot k, \quad (11)$$

where $b(\mathcal{E}_{r\text{DS}}, g, r \cdot 2^{O(h \log h)}, \mathcal{G})$ is the function defined in Lemma 3. \square

It can be easily checked that the multiplicative constant involved in the upper bound of Equation (11) is $2^{2^{2^{r \cdot \log r} \cdot 2^{O(h \cdot \log h)}}}$, that is, it depends triple-exponentially on the integer r .

5 An explicit linear kernel for r -Scattered Set

Let $r \geq 1$ be a fixed integer. Given a graph G and a set $S \subseteq V(G)$, we say that S is an r -independent set if any two vertices in S are at distance greater than r in G . We define the r -SCATTERED SET problem, which can be seen as a generalization of INDEPENDENT SET, as follows.

r -SCATTERED SET

Instance: A graph G and a non-negative integer k .

Parameter: The integer k .

Question: Does G have a $2r$ -independent set of size at least k ?

Our encoder for r -SCATTERED SET (or equivalently, for r -INDEPENDENT SET) is inspired from the proof of Fomin *et al.* [7] that the problem has FII, and can be found in Subsection 5.1. We then show how to construct the linear kernel in Subsection 5.2.

5.1 Description of the encoder

Equivalently, we proceed to present an encoder for the r -INDEPENDENT SET problem, which we abbreviate as $r\text{IS}$. Let G be a boundaried graph with boundary $\partial(G)$ and denote $I = \Lambda(G)$. The function $\mathcal{C}_{r\text{IS}}$ maps I to a set $\mathcal{C}_{r\text{IS}}(I)$ of $\mathcal{C}_{r\text{IS}}$ -encodings. Each $R \in \mathcal{C}_{r\text{IS}}(I)$ maps I to an $|I|$ -tuple the coordinates of which are in one-to-one correspondence with the vertices of $\partial(G)$. The coordinate $R(v)$ of vertex $v \in \partial(G)$ is a $(|I| + 1)$ -tuple in $(d_S, d_{v_1}, \dots, d_{v_{|I|}}) \in \{0, 1, \dots, r, r+1\}^{|I|+1}$. For a subset S of vertices of G , we say that (G, S, R) belongs to the language $L_{\mathcal{C}_{r\text{IS}}}$ (or that S is a *partial r -independent set satisfying R*) if:

- for every pair of vertices $v \in S$ and $w \in S$, $d_G(v, w) > r$;
- for every vertex $v \in \partial(G)$: $d_G(v, S) \geq d_S$ and for every $w \in \partial(G)$, $d_G(v, w) \geq d_w$.

As r -INDEPENDENT SET is a maximization problem, by Equation (2) the function $f_G^{\mathcal{E}_{r\text{IS}}}$ associates to each encoding R the maximum size of a partial r -independent set S satisfying R . By definition of $\mathcal{E}_{r\text{IS}}$ it is clear that

$$s_{\mathcal{E}_{r\text{IS}}}(t) \leq (r+2)^{t(t+1)}. \quad (12)$$

Lemma 6. *The encoder $\mathcal{E}_{r\text{IS}} = (\mathcal{C}_{r\text{IS}}, L_{\mathcal{C}_{r\text{IS}}})$ described above is an $r\text{IS}$ -encoder. Furthermore, if \mathcal{G} is an arbitrary class of graphs and $g(t) = 2t$, then the equivalence relation $\sim_{\mathcal{E}_{r\text{IS}}, g, t, \mathcal{G}}$ is DP-friendly.*

Proof: We first prove that $\mathcal{E}_{r\text{IS}} = (\mathcal{C}_{r\text{IS}}, L_{\mathcal{C}_{r\text{IS}}})$ is a $r\text{IS}$ -encoder. There is a unique 0-tuple $R_\emptyset \in \mathcal{C}_{r\text{IS}}(\emptyset)$, and by definition of $L_{\mathcal{C}_{r\text{IS}}}$, $(G, S, R_\emptyset) \in L_{\mathcal{C}_{r\text{IS}}}$ if and only if S is an r -independent set of G .

Let G, G' with boundary A and H, G_B, G'_B with boundary B be the graphs as defined in the proof of Lemma 5 (see Fig. 2).

Let R_0 be the encoding satisfying $R_0(v) = (0, 0, \dots, 0)$ for every $v \in B$. Observe that if S is a maximum partial r -independent set satisfying an encoding $R_B \in \mathcal{C}_{r\text{IS}}(\Lambda(G_B))$, then S also satisfies R_0 . It follows that $f_G^{\mathcal{E}_{r\text{IS}}}(R_0) = \max_{R_B} f_G^{\mathcal{E}_{r\text{IS}}}(R_B)$ (and thus $f_G^{\mathcal{E}_{r\text{IS}}}(R_0) = f_G^{\mathcal{E}_{r\text{IS}},g}(R_0)$).

We want to show that $G \sim_{\mathcal{E}_{r\text{IS}},g,t} G'$ and that $\Delta_{\mathcal{E}_{r\text{IS}},g,t}(G, G') = \Delta_{\mathcal{E}_{r\text{IS}},g,t}(G_B, G'_B)$. According to Fact 1, it is enough to consider the relation $\sim_{\mathcal{E}_{r\text{IS}},g,t}$. To that aim, we will prove that $f_G^{\mathcal{E}_{r\text{IS}},g}(R_A) = f_{G'}^{\mathcal{E}_{r\text{IS}},g}(R_A) + \Delta_{\mathcal{E}_{r\text{IS}},g,t}(G_B, G'_B)$ for all $R_A \in \mathcal{C}_{r\text{IS}}(\Lambda(G))$ and for $g(t) = 2t$.

Let $R_A \in \mathcal{C}_{r\text{IS}}(\Lambda(G))$ be a $\mathcal{C}_{r\text{IS}}$ -encoding defined on A . First assume that $f_G^{\mathcal{E}_{r\text{IS}},g}(R_A) \neq -\infty$, that is, $R_A \in \mathcal{C}_{r\text{IS},G}^*$. Let $S = I \cup I_H$ be a partial r -independent set of size $f_G^{\mathcal{E}_{r\text{IS}},g}(R_A)$ of G , with $I \subseteq V(G_B)$ and $I_H \subseteq V(H) \setminus B$. An encoding $R_B \in \mathcal{C}_{r\text{IS}}(\Lambda(G_B))$, satisfied by S is defined as follows. Let $v \in B$, then $R_B(v) = (d_S, d_{v_1}, \dots, d_{v_{|B|}})$ where

- $d_S = d_{G_B}(v, I)$; and
- for $i \in \{1, \dots, |B|\}$, $d_{v_i} = \min\{d_{G_B}(v, v_i), r + 1\}$ (remind that $v_i \in \partial(G)$).

Fact 2 For the R_B defined above, it holds that $f_{G_B}^{\mathcal{E}_{r\text{IS}},g}(R) \neq -\infty$, where $g(t) = 2t$.

Proof: Let $I_0 \subseteq V(G_B)$ be a maximum partial r -independent set satisfying R_0 , i.e., $(G_B, I_0, R_0) \in L_{\mathcal{C}_{r\text{IS}}}$. Let us define $I^* = I \setminus N_{r/2}(B)$, $I_0^* = I_0 \setminus N_{r/2}(B)$ and $I_H^* = I_H \setminus N_{r/2}(B)$. By the pigeon-hole principle, it is easy to see that $|I_0^*| \geq |I_0| - t$ (otherwise B would contain a vertex at distance at most $r/2$ from two distinct vertices of I_0). Likewise, $|I_H^*| \geq |I_H| - t$. Now observe that $I_0^* \cup I_H^*$ is an r -independent set of G and therefore $|I_0^*| + |I_H^*| \leq |S|$ (1) (as S was chosen as a maximum r -independent set of G). As S is the disjoint union of I and I_H , we also have that $|S| \leq |I| + |I_H^*| + t$ (2). Combining (1) and (2), we obtain that $|I_0^*| \leq |I| + t$ and therefore $|I_0| \leq |I| + 2t$. It follows that $f_{G_B}^{\mathcal{E}_{r\text{IS}}}(R_B) = f_{G_B}^{\mathcal{E}_{r\text{IS}},g}(R_B)$, proving the fact. \square

Observe that by construction of R_B , $|I| \leq f_{G_B}^{\mathcal{E}_{r\text{IS}},g}(R_B)$. Consider a subset of vertices I' of G'_B of maximum size such that $(G'_B, I', R_B) \in L_{\mathcal{C}_{r\text{IS}}}$, that is $|I'| = f_{G'_B}^{\mathcal{E}_{r\text{IS}},g}(R_B)$. As $G_B \equiv_{\mathcal{E}_{r\text{IS}},t} G'_B$, by the above claim, we have $|I'| = f_{G_B}^{\mathcal{E}_{r\text{IS}},g}(R_B) + \Delta_{\mathcal{E}_{r\text{IS}},g,t}(G_B, G'_B)$ and therefore $|I' \cup I_H| = f_{G_B}^{\mathcal{E}_{r\text{IS}},g}(R_B) + \Delta_{\mathcal{E}_{r\text{IS}},g,t}(G_B, G'_B) + |I_H| \geq f_G^{\mathcal{E}_{r\text{IS}},g}(R_A) + \Delta_{\mathcal{E}_{r\text{IS}},g,t}(G_B, G'_B)$.

Let us prove that $S' = I' \cup I_H$ is a partial r -independent set of G' satisfying R_A . Following the definition of $\mathcal{E}_{r\text{IS}}$, we have to verify two kinds of conditions: those on vertices in S' and those on vertices in A . We start with vertices in S' . Let P be a shortest path in G between two vertices $v \in S'$ and $w \in S'$. We partition P into maximal subpaths P_1, \dots, P_q such that P_j (for $1 \leq j \leq q$) is either a path of G'_B (called a G'_B -path) or of H (called an H -path). An illustration of these paths can be found in Fig. 4. If $q = 1$, then $d_{G'}(v, w) > r$ follows from the fact that I_H and I' are respectively r -independent sets of H and G'_B (a partial r -independent set is an r -independent set). So assume that $q > 1$. Observe that every H -subpath is a path in G . By the choice of S' , observe that the length of every G'_B -subpath is at least the distance in G_B between its extremities. We consider three cases:

- $v, w \in V(H) \setminus B$: By the observations above, the length of P is at least $d_G(v, w)$. As $v, w \in I_H$, we obtained that $d_{G'}(v, w) \geq d_G(v, w) > r$.

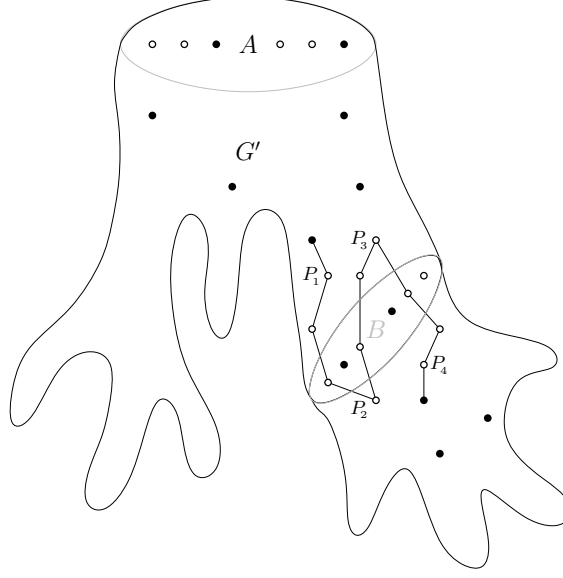


Fig. 4. Illustration in the proof of Lemma 6. The black vertices belong to the solution.

- $v \in V(H) \setminus B$ and $w \in V(G'_B)$: Let u be the last vertex of P_{q-1} . By the same argument as in the previous case we have $d_{G'}(v, u) \geq d_G(v, u)$. Now by the choice of S' , observe that $d_{G'_B}(u, w) \geq d_{G_B}(u, I)$. So the length of P is at least the distance in G from v to a vertex $w' \in I$, we can conclude that $d_{G'}(v, w) > r$.
- $v, w \in V(G'_B)$: Let u_1 and u_q be respectively the last vertex of P_1 and the first vertex of P_q . By the same argument as above, we have that $d_{G'}(u_1, u_q) \geq d_G(u_1, u_q)$. By the choice of S' , we have that $d_{G'_B}(u_1, v) \geq d_{G_B}(u_1, I)$ and $d_{G'_B}(u_q, w) \geq d_{G_B}(u_q, I)$. So the length of P is at least the distance in G between two vertices $v' \in I$ and $w' \in I$. We can therefore conclude that $d_{G'}(v, w) > r$.

We now consider vertices of A . Let $v \in A$ such that $R_A(v) = (d_S, d_{v_1}, \dots, d_{v_{|A|}})$. Let P be a shortest path in G' between vertices $v \in A$ and $w \in S'$, similarly to the previous argumentation (two first items) $d_{G'}(v, w) > d_S$. Now let P be a shortest path in G' between vertices $v \in A$ and $v_i \in A$ similarly to the previous argumentation (first item) $d_{G'}(v, w) > d_{v_i}$.

It follows that $S' = I' \cup I_H$ is a partial r -independent set of size at least $f_G^{\mathcal{E}_{rIS}, g}(R_A) + \Delta_{\mathcal{E}_{rIS}, g, t}(G_B, G'_B)$ satisfying R_A , as we wanted to prove.

Finally, assume that $f_G^{\mathcal{E}_{rIS}}(R_A) = -\infty$. Then it holds that $f_{G'}^{\mathcal{E}_{rIS}}(R_A) = -\infty$ as well. Indeed, suppose that $f_{G'}^{\mathcal{E}_{rIS}}(R_A)$ is finite. Then, given a partial r -independent set of G' satisfying R_A , by the argument above we could construct a partial r -independent set of G satisfying R_A , contradicting that $f_G^{\mathcal{E}_{rIS}}(R_A) = -\infty$.

Therefore, we can conclude that $G \sim_{\mathcal{E}_{rIS}, g, t} G'$, and hence the equivalence relation $\sim_{\mathcal{E}_{rIS}, g, t, G}$ is DP-friendly for $g(t) = 2t$. \square

5.2 Construction of the kernel

For constructing a linear kernel, we use the following observation, also noted in [7]. Suppose that (G, k) is a NO-instance of r -SCATTERED SET. Then, if for $1 \leq i \leq k$ we

greedily choose a vertex v_i in $G - \bigcup_{j < i} N_{2r}[v_j]$, the graph $G - \bigcup_{1 \leq i \leq k} N_{2r}[v_i]$ is empty. Thus, $\{v_1, \dots, v_k\}$ is a $2r$ -dominating set.

Lemma 7 (Fomin *et al.* [7]). *If (G, k) is a NO-instance of the r -SCATTERED SET problem, then (G, k) is a YES-instance of the $2r$ -DOMINATING SET problem.*

We are ready to present the linear kernel for r -SCATTERED SET on apex-minor-free graphs.

Theorem 5. *Let $r \geq 1$ be an integer, let H be an h -vertex apex graph, and let rSS_H be the restriction of the r -SCATTERED SET problem to input graphs which exclude H as a minor. Then rSS_H admits a constructive linear kernel of size at most $f(r, h) \cdot k$, where f is an explicit function depending only on r and h , defined in Equation (13) below.*

Proof: Given an instance (G, k) of rSS_H , we run on it the algorithm given by Theorem 3 for the r' -DOMINATING SET problem with $r' := 2r$. If the algorithm is not able to find a set X of the claimed size, then by Lemma 7 we can conclude that $(G, k) \in rSS_H$. Otherwise, we use again the set X as input to the algorithm given by Theorem 2, which outputs in linear time a $(r^2 \cdot 2^{O(h \log h)} \cdot k, r \cdot 2^{O(h \log h)})$ -protrusion decomposition of G . We now consider the encoder $\mathcal{E}_{rIS} = (\mathcal{C}_{rIS}, L_{\mathcal{C}_{rIS}})$ defined in Subsection 5.1. By Lemma 6, \mathcal{E}_{rIS} is an rIS -encoder and $\sim_{\mathcal{E}_{rIS}, g, t, \mathcal{G}}$ is DP-friendly, where \mathcal{G} is the class of H -minor-free graphs and $g(t) = 2t$, and furthermore by Equation (12) it satisfies $s_{\mathcal{E}_{rIS}}(t) \leq (r+2)^{t(t+1)}$. Therefore, we are again in position to apply Corollary 1 and obtain a linear kernel for rSS_H of size at most

$$r^2 \cdot 2^{O(h \log h)} \cdot b\left(\mathcal{E}_{rIS}, g, r \cdot 2^{O(h \log h)}, \mathcal{G}\right) \cdot k, \quad (13)$$

where $b(\mathcal{E}_{rIS}, g, r \cdot 2^{O(h \log h)}, \mathcal{G})$ is the function defined in Lemma 3. \square

6 An explicit linear kernel for Planar- \mathcal{F} -Deletion

Let \mathcal{F} be a finite set of graphs. We define the \mathcal{F} -DELETION problem as follows.

\mathcal{F} -DELETION

Instance: A graph G and a non-negative integer k .

Parameter: The integer k .

Question: Does G have a set $S \subseteq V(G)$ such that $|S| \leq k$ and $G - S$ is H -minor-free for every $H \in \mathcal{F}$?

When all the graphs in \mathcal{F} are connected, the corresponding problem is called CONNECTED- \mathcal{F} -DELETION, and when \mathcal{F} contains at least one planar graph, we call it PLANAR- \mathcal{F} -DELETION. When both conditions are satisfied, the problem is called CONNECTED-PLANAR- \mathcal{F} -DELETION. Note that CONNECTED-PLANAR- \mathcal{F} -DELETION encompasses, in particular, VERTEX COVER and FEEDBACK VERTEX SET.

Our encoder for the \mathcal{F} -DELETION problem uses the dynamic programming machinery developed by Adler *et al.* [1], and it is described in Subsection 6.1. The properties of this encoder also guarantee that the equivalence relation $\sim_{\mathcal{G}, t}$ has finite index (see the last paragraph of Subsection 3.3). We prove that this encoder is indeed an \mathcal{F} -DELETION-encoder and that the corresponding equivalence relation is DP-friendly, under the constraint that all the graphs in \mathcal{F} are *connected*. Interestingly, this phenomenon concerning

the connectivity seems to be in strong connection with the fact that the \mathcal{F} -DELETION problem has FII if all the graphs in \mathcal{F} are connected [7, 17], but for some families \mathcal{F} containing disconnected graphs, \mathcal{F} -DELETION has not FII (see [22] for an example of such family).

We then obtain a linear kernel for the problem using two different approaches. The first one, described in Subsection 6.1, follows the same scheme as the one used in the previous sections (Sections 4 and 5), that is, we first find a treewidth-modulator X in polynomial time, and then we use this set X as input to the algorithm of Theorem 2 to find a linear protrusion decomposition of the input graph. In order to find the treewidth-modulator X , we need that the input graph G excludes a fixed graph H as a minor.

With our second approach, which can be found in Subsection 6.3, we obtain a linear kernel on the larger class of graphs that exclude a fixed graph H as a *topological* minor. We provide two variants of this second approach. One possibility is to use the randomized constant-factor approximation for PLANAR- \mathcal{F} -DELETION by Fomin *et al.* [17] as treewidth-modulator, which yields a randomized linear kernel that can be found in uniform polynomial time. The second possibility consists in arguing just about the *existence* of a linear protrusion decomposition in YES-instances, and then greedily finding large protrusions to be reduced by the protrusion replacer given by Theorem 1. This yields a deterministic linear kernel that can be found in time $n^{f(H, \mathcal{F})}$, where f is a function depending on H and \mathcal{F} .

6.1 The encoder for \mathcal{F} -Deletion and the index of $\sim_{\mathcal{G}, t}$

In this subsection we define an encoder $\mathcal{E}_{\mathcal{F}\text{D}} = (\mathcal{C}_{\mathcal{F}\text{D}}, L_{\mathcal{C}_{\mathcal{F}\text{D}}})$ for \mathcal{F} -DELETION, and along the way we will also prove that when \mathcal{G} is the class of graphs excluding a fixed graph on h vertices as a minor, then the index of the equivalence relation $\sim_{\mathcal{G}, t}$ is bounded by $2^{t \log t} \cdot h^t \cdot 2^{h^2}$.

Recall first that a *model* of a graph F in a graph G is a mapping ϕ , that assigns to every edge $e \in E(F)$ an edge $\phi(e) \in E(G)$, and to every vertex $v \in V(F)$ a non-empty connected subgraph $\phi(v) \subseteq G$, such that

- (i) the graphs $\{\phi(v) \mid v \in V(F)\}$ are mutually vertex-disjoint and the edges $\{\phi(e) \mid e \in E(F)\}$ are pairwise distinct;
- (ii) for $e = \{u, v\} \in E(F)$, $\phi(e)$ has one end-vertex in $V(\phi(u))$ and the other in $V(\phi(v))$.

Assume first for simplicity that $\mathcal{F} = \{F\}$ consists of a single connected graph F . Following [1], we introduce a combinatorial object called *rooted packing*. These objects are originally defined for branch decompositions, but we can directly translate them to tree decompositions. Loosely speaking, rooted packings capture how “potential models” of F intersect the separators that the algorithm is processing. It is worth mentioning that the notion of rooted packing is related to the notion of *folio* introduced by Robertson and Seymour in [27], but more suited to dynamic programming. See [1] for more details.

Formally, let $S_F^* \subseteq V(F)$ be a subset of the vertices of the graph F , and let $S_F \subseteq S_F^*$. Given a bag B of a tree decomposition (T, \mathcal{X}) of the input graph G , we define a *rooted packing* of B as a quintuple $\mathbf{rp} = (\mathcal{A}, S_F^*, S_F, \psi, \chi)$, where \mathcal{A} is a (possible empty) collection of mutually disjoint non-empty subsets of B (that is, a *packing* of B), $\psi : \mathcal{A} \rightarrow S_F$ is a surjective mapping (called the *rooting*) assigning vertices of S_F to the sets in \mathcal{A} , and $\chi : S_F \times S_F \rightarrow \{0, 1\}$ is a binary symmetric function between pairs of vertices in S_F .

The intended meaning of a rooted packing $(\mathcal{A}, S_F^*, S_F, \psi, \chi)$ is as follows. In a given separator B , a packing \mathcal{A} represents the intersection of the connected components of the potential model with B . The subsets $S_F^*, S_F \subseteq V(F)$ and the function χ indicate

that we are looking in the graph G_B for a potential model of $F[S_F^*]$ containing the edges between vertices in S_F given by the function χ . Namely, the function χ captures which edges of $F[S_F^*]$ have been realized so far in the processed graph. Since we allow the vertex-models intersecting B to be disconnected, we need to keep track of their connected components. The subset $S_F \subseteq S_F^*$ tells us which vertex-models intersect B , and the function ψ associates the sets in \mathcal{A} with the vertices in S_F . We can think of ψ as a coloring that colors the subsets in \mathcal{A} with colors given by the vertices in S_F . Note that several subsets in \mathcal{A} can have the same color $u \in S_F$, which means that the vertex-model of u in G_B is not connected yet, but it may get connected in further steps of the dynamic programming. Again, see [1] for the details.

It is proved in [1] that rooted packings allow to carry out dynamic programming in order to determine whether an input graph G contains a graph F as a minor. It is easy to see that the number of distinct rooted packings at a bag B is upper-bounded by $f(t, F) := 2^{t \log t} \cdot r^t \cdot 2^{r^2}$, where $t = \text{tw}(G)$ and $r = |V(F)|$. In particular, this proves that when \mathcal{G} is the class of graphs excluding a fixed graph H on h vertices as a minor, then the index of the equivalence relation $\sim_{\mathcal{G}, t}$ is bounded by $2^{t \log t} \cdot h^t \cdot 2^{h^2}$.

Nevertheless, in order to solve the \mathcal{F} -DELETION problem, we need a more complicated data structure. The intuitive reason is that it is inherently more difficult to cover *all* models of a graph F with at most k vertices, rather than just finding one. We define $\mathcal{C}_{\mathcal{F}\text{D}}$ as the function which maps $I \subseteq \{1, \dots, t\}$ to a subspace of $\{0, 1\}^{f(|I|, F)}$. That is, each $\mathcal{C}_{\mathcal{F}\text{D}}$ -encoding $R \in \mathcal{C}(I)$ is a vector of $f(|I|, F)$ bits, which when interpreted as the tables of a dynamic programming algorithm at a given bag B such that $\Lambda(G_B) = I$, prescribes which rooted packings exist in the graph G_B once the corresponding vertices of the desired solution to \mathcal{F} -DELETION have been removed. More precisely, the language $L_{\mathcal{C}_{\mathcal{F}\text{D}}}$ contains the triples (G, S, R) (recall from Definition 6 that here G is a boundaried graph with $\Lambda(G) \subseteq I$, $S \subseteq V(G)$, and $R \in \mathcal{C}(I)$) such that the graph $G - S$ contains precisely the rooted packings prescribed by R (namely, those whose corresponding bit equals 1 in R), and such that the graph $G - (\partial G \cup S)$ does *not* contain F as a minor.

When the family $\mathcal{F} = \{F_1, \dots, F_\ell\}$ may contain more than one graph, let $f(t, \mathcal{F}) = \sum_{i=1}^{\ell} f(t, F_i)$, and we define $\mathcal{C}_{\mathcal{F}\text{D}}$ as the function which maps $I \subseteq \{1, \dots, t\}$ to a subspace of $\{0, 1\}^{f(|I|, \mathcal{F})}$. The language is defined $L_{\mathcal{C}_{\mathcal{F}\text{D}}}$ is defined accordingly, that is, such that the graph $G - S$ contains precisely the rooted packings of F_i prescribed by R , for each $1 \leq i \leq \ell$, and such that the graph $G - (\partial G \cup S)$ does *not* contain any of the graphs in \mathcal{F} as a minor. By definition of $\mathcal{E}_{\mathcal{F}\text{D}}$, it clearly holds that

$$s_{\mathcal{E}_{\mathcal{F}\text{D}}}(t) \leq 2^{f(t, F_1)} \cdot 2^{f(t, F_2)} \dots 2^{f(t, F_\ell)} = 2^{f(t, \mathcal{F})}. \quad (14)$$

Assume henceforth that all graphs in the family \mathcal{F} are *connected*. This assumption is crucial because for a connected graph $F \in \mathcal{F}$ and a potential solution S , as the graph $G - (\partial G \cup S)$ does not contain F as a minor, we can assume that the packing \mathcal{A} corresponding to a potential model of F rooted at $\partial G \setminus S$ is *nonempty*. Indeed, as F is connected, a rooted packing which does *not* intersect $\partial G \setminus S$ can never be extended to a (complete) model of F in $G \oplus K$ for any t -boundaried graph K . Therefore, we can directly discard these empty rooted packings. We will use this property in the proof of Lemma 8 below. Note that this assumption is not safe if F contains more than one connected component. As mentioned before, this phenomenon seems to be in strong connection with the fact that the \mathcal{F} -DELETION problem has FII if all the graphs in \mathcal{F} are connected [7, 17], but for some families \mathcal{F} containing disconnected graphs, \mathcal{F} -DELETION has not FII.

Lemma 8. *The encoder $\mathcal{E}_{\mathcal{F}\text{D}}$ is a CONNECTED- \mathcal{F} -DELETION-encoder. Furthermore, if \mathcal{G} is an arbitrary class of graphs and $g(t) = t$, then the equivalence relation $\sim_{\mathcal{E}_{\mathcal{F}\text{D}}, g, t, \mathcal{G}}$ is DP-friendly.*

Proof: The fact that $\mathcal{E}_{\mathcal{F}\mathcal{D}} = (\mathcal{C}_{\mathcal{F}\mathcal{D}}, L_{\mathcal{C}_{\mathcal{F}\mathcal{D}}})$ is a CONNECTED- \mathcal{F} -DELETION-encoder follows easily from the above discussion, as if G is a 0-boundaried graph, then $\mathcal{C}_{\mathcal{F}\mathcal{D}}(\emptyset)$ consists of a single $\mathcal{C}_{\mathcal{F}\mathcal{D}}$ -encoding R_\emptyset , and $(G, S, R_\emptyset) \in L_{\mathcal{C}_{\mathcal{F}\mathcal{D}}}$ if and only if the graph $G - S$ contains none of the graphs in \mathcal{F} as a minor. It remains to prove that the equivalence relation $\sim_{\mathcal{E}_{\mathcal{F}\mathcal{D}}, g, t, \mathcal{G}}$ is DP-friendly for $g(t) = t$.

The proof is similar to the proofs for r -DOMINATING SET and r -SCATTERED SET, so we will omit some details. As in the proof of Lemma 5, we start by proving that the encoder $\mathcal{E}_{\mathcal{F}\mathcal{D}}$ for CONNECTED- \mathcal{F} -DELETION is g -confined for the identity function $g(t) = t$. Similarly to the encoder we presented for r -DOMINATING SET, $\mathcal{E}_{\mathcal{F}\mathcal{D}} = (\mathcal{C}_{\mathcal{F}\mathcal{D}}, L_{\mathcal{C}_{\mathcal{F}\mathcal{D}}})$ has the following monotonicity property. For $R_1, R_2 \in \mathcal{C}_{\mathcal{F}}(I)$ such that $f_G^{\mathcal{C}_{\mathcal{F}\mathcal{D}}}(R_1) < \infty$ and $f_G^{\mathcal{C}_{\mathcal{F}\mathcal{D}}}(R_2) < \infty$,

$$\text{if } R_1^{-1}(0) \subseteq R_2^{-1}(0), \text{ then } f_G^{\mathcal{C}_{\mathcal{F}\mathcal{D}}}(R_1) \leq f_G^{\mathcal{C}_{\mathcal{F}\mathcal{D}}}(R_2), \quad (15)$$

where for $i \in \{1, 2\}$, $R_i^{-1}(0)$ denotes the set of rooted packings whose corresponding bit equals 0 in R_i . Indeed, Equation (15) holds because any solution S in G that covers all the rooted packings forbidden by R_2 also covers those forbidden by R_1 (as by hypothesis $R_1^{-1}(0) \subseteq R_2^{-1}(0)$), so it holds that $f_G^{\mathcal{C}_{\mathcal{F}\mathcal{D}}}(R_1) \leq f_G^{\mathcal{C}_{\mathcal{F}\mathcal{D}}}(R_2)$. Let $R_0 = \{0, 0, \dots, 0\}$ be the $\mathcal{C}_{\mathcal{F}\mathcal{D}}(I)$ -encoding with all the bits set to 0. The key observation is that, since each graph in \mathcal{F} is *connected*, by the discussion above the lemma we can assume that each packing \mathcal{A} in a rooted packing is nonempty. This implies that if $R \in \mathcal{C}_{\mathcal{F}\mathcal{D}}(I)$ such that $(G, S, R) \in L_{\mathcal{C}_{\mathcal{F}\mathcal{D}}}$ for some set $S \subseteq V(G)$, then $(G, S \cup \partial G, R_0) \in L_{\mathcal{C}_{\mathcal{F}}}$. In other words, any solution S for an arbitrary $\mathcal{C}_{\mathcal{F}\mathcal{D}}$ -encoder R can be transformed into a solution for R_0 by adding a set of vertices of size at most $|\partial(G)| \leq t$. As by Equation (15), for any $\mathcal{C}_{\mathcal{F}\mathcal{D}}$ -encoding R with $f_G^{\mathcal{C}_{\mathcal{F}\mathcal{D}}}(R) < \infty$, it holds that $f_G^{\mathcal{C}_{\mathcal{F}\mathcal{D}}}(R) \leq f_G^{\mathcal{C}_{\mathcal{F}\mathcal{D}}}(R_0)$, it follows that for any graph G with $\Lambda(G) = I$,

$$\max_{R \in \mathcal{C}_{\mathcal{F}\mathcal{D}, G}^*(I)} f_G^{\mathcal{E}_{\mathcal{F}\mathcal{D}}}(R) - \min_{R \in \mathcal{C}_{\mathcal{F}\mathcal{D}, G}^*(I)} f_G^{\mathcal{E}_{\mathcal{F}\mathcal{D}}}(R) \leq t, \quad \text{as we wanted to prove.}$$

Once we have that $\mathcal{E}_{\mathcal{F}\mathcal{D}} = (\mathcal{C}_{\mathcal{F}\mathcal{D}}, L_{\mathcal{C}_{\mathcal{F}\mathcal{D}}})$ is g -confined, the proof goes along the same lines of that of Lemma 5. That is, the objective is to show that, in the setting depicted in Fig. 2, $G \sim_{\mathcal{E}_{\mathcal{F}\mathcal{D}}, g, t, \mathcal{G}} G'$ (due to Fact 1) and $\Delta_{\mathcal{E}_{\mathcal{F}\mathcal{D}}, g, t}(G, G') = \Delta_{\mathcal{E}_{\mathcal{F}\mathcal{D}}, g, t}(G_B, G'_B)$. Due to the g -confinement, it suffices to prove that $f_G^{\mathcal{E}_{\mathcal{F}\mathcal{D}}}(R_A) = f_{G'}^{\mathcal{E}_{\mathcal{F}\mathcal{D}}}(R_A) + \Delta_{\mathcal{E}_{\mathcal{F}\mathcal{D}}, g, t}(G_B, G'_B)$ for all $R_A \in \mathcal{C}_{\mathcal{F}\mathcal{D}}(\Lambda(G))$. Since $G_B \sim_{\mathcal{E}_{\mathcal{F}\mathcal{D}}, g, t} G'_B$, the definition of $\mathcal{E}_{\mathcal{F}\mathcal{D}}$ it implies that the graphs G_B and G'_B contain exactly the same set of rooted packings, so their behavior with respect to H (see Fig. 2) in terms of the existence of models of graphs in \mathcal{F} is exactly the same. For more details, it is proved in [1] that using the encoder $\mathcal{E}_{\mathcal{F}\mathcal{D}} = (\mathcal{C}_{\mathcal{F}\mathcal{D}}, L_{\mathcal{C}_{\mathcal{F}\mathcal{D}}})$, the tables of a given bag in a tree- or branch-decomposition can indeed be computed from the tables of their children. Therefore, we have that $G \sim_{\mathcal{E}_{\mathcal{F}\mathcal{D}}, g, t, \mathcal{G}} G'$. Finally, the fact that $f_G^{\mathcal{E}_{\mathcal{F}\mathcal{D}}}(R_A) = f_{G'}^{\mathcal{E}_{\mathcal{F}\mathcal{D}}}(R_A) + \Delta_{\mathcal{E}_{\mathcal{F}\mathcal{D}}, g, t}(G_B, G'_B)$ can be easily proved by noting that any set $S \in V(G)$ satisfying R_A can be transformed into a set $S' \in V(G')$ satisfying R_A such that $|S'| \leq |S| - \Delta_{\mathcal{E}_{\mathcal{F}\mathcal{D}}, g, t}(G_B, G'_B)$ (by just replacing $S \cap V(G_B)$ with the corresponding set of vertices in $V(G'_B)$, using that $G_B \sim_{\mathcal{E}_{\mathcal{F}\mathcal{D}}, g, t} G'_B$), and vice versa. \square

6.2 Construction of the kernel on H -minor-free graphs

The objective of this subsection is to prove the following theorem.

Theorem 6. *Let \mathcal{F} be a finite set of connected graphs containing at least one r -vertex planar graph F , let H be an h -vertex graph, and let CPFD_H be the restriction of the*

CONNECTED-PLANAR- \mathcal{F} -DELETION problem to input graphs which exclude H as a minor. Then CPFD_H admits a constructive linear kernel of size at most $f(r, h) \cdot k$, where f is an explicit function depending only on r and h , defined in Equation (16) below.

We first need a powerful theorem, which follows mainly from the results of Fomin *et al.* [18], and uses also [20, 21].

Theorem 7 (Fomin *et al.* [18]). *Let \mathcal{F} be a finite set of graphs containing at least one r -vertex planar graph F , let H be an h -vertex graph, and let PFD_H be the restriction of the PLANAR- \mathcal{F} -DELETION problem to input graphs which exclude H as a minor. If $(G, k) \in \text{PFD}_H$, then there exists a set $X \subseteq V(G)$ such that $|X| = r \cdot 2^{O(h^2)} \cdot k$ and $\text{tw}(G - X) = r \cdot 2^{O(h^2)}$. Moreover, given an instance (G, k) of PFD_H with $|V(G)| = n$, there is an algorithm running in time $O(n^3)$ that either finds such a set X or correctly reports that (G, k) is a NO-instance.*

We are ready to present a linear kernel for CONNECTED-PLANAR- \mathcal{F} -DELETION when the input graph excludes a fixed graph H as a minor.

Proof of Theorem 6: The proof is very similar to the one of Theorem 4. Given an instance (G, k) , we run the cubic algorithm given by Theorem 7 to either conclude that (G, k) is a NO-instance or to find a set $X \subseteq V(G)$ such that $|X| = r \cdot 2^{O(h^2)} \cdot k$ and $\text{tw}(G - X) = r \cdot 2^{O(h^2)}$. In the latter case, we use the set X as input to the algorithm given by Theorem 2, which outputs in linear time a $(r^2 \cdot 2^{O(h^2)} \cdot k, r \cdot 2^{O(h^2)})$ -protrusion decomposition of G . We now consider the encoder $\mathcal{E}_{\mathcal{F}\mathcal{D}} = (\mathcal{C}_{\mathcal{F}\mathcal{D}}, L_{\mathcal{C}_{\mathcal{F}\mathcal{D}}})$ defined in Subsection 6.1. By Lemma 8, $\mathcal{E}_{\mathcal{F}\mathcal{D}}$ is a CPFD_H -encoder and $\sim_{\mathcal{E}_{\mathcal{F}\mathcal{D}}, g, t, \mathcal{G}}$ is DP-friendly, where $g(t) = t$ and \mathcal{G} is the class of H -minor-free graphs. An upper bound on $s_{\mathcal{E}_{\mathcal{F}\mathcal{D}}}(t)$ is given in Equation (14). Therefore, we are in position to apply Corollary 1 and obtain a linear kernel for CPFD_H of size at most

$$r^2 \cdot 2^{O(h^2)} \cdot b\left(\mathcal{E}_{\mathcal{F}\mathcal{D}}, g, r \cdot 2^{O(h^2)}, \mathcal{G}\right) \cdot k, \quad (16)$$

where $b\left(\mathcal{E}_{\mathcal{F}\mathcal{D}}, g, r \cdot 2^{O(h^2)}, \mathcal{G}\right)$ is the function defined in Lemma 3. \square

6.3 Linear kernels on H -topological-minor-free graphs

In this subsection we explain how to obtain linear kernels for PLANAR- \mathcal{F} -DELETION on graphs excluding a topological minor. We first describe a uniform randomized kernel and then a nonuniform deterministic one. We would like to note that in the case that \mathcal{G} is the class of graphs excluding a fixed h -vertex graph H as a topological minor, by using a slight variation of the rooted packings described in Subsection 6.1 it can be proved, using standard dynamic techniques, that the index of the equivalence relation $\sim_{\mathcal{G}, t}$ is also upper-bounded by $2^{t \log t} \cdot h^t \cdot 2^{h^2}$.

Before presenting the uniform randomized kernel, we need the following two results.

Theorem 8 (Fomin *et al.* [17]). *The optimization version of the PLANAR- \mathcal{F} -DELETION problem admits a randomized constant-factor approximation.*

Theorem 9 (Leaf and Seymour [24]). *For every simple planar graph F on r vertices, every F -minor-free graph G satisfies $\text{tw}(G) \leq 2^{15r+8r \log r}$.*

Theorem 10. *Let \mathcal{F} be a finite set of connected graphs containing at least one r -vertex planar graph F , let H be an h -vertex graph, and let $\text{CPFD}_{H\text{-top}}$ be the restriction of the CONNECTED-PLANAR- \mathcal{F} -DELETION problem to input graphs which exclude H as a topological minor. Then $\text{CPFD}_{H\text{-top}}$ admits a linear randomized kernel of size at most $f(r, h) \cdot k$, where f is an explicit function depending only on r and h , defined in Equation (17) below.*

Proof: Given an instance (G, k) of $\text{CPFD}_{H\text{-top}}$, we first run the randomized polynomial-time approximation algorithm given by Theorem 8, which achieves an expected constant ratio $c_{\mathcal{F}}$. If we obtain a solution $X \subseteq V(G)$ such that $|X| > c_{\mathcal{F}} \cdot k$, we declare that (G, k) is a NO-instance. Otherwise, if $|X| \leq c_{\mathcal{F}} \cdot k$, we use the set X as input to the algorithm given by Theorem 2. As by Theorem 9 we have that $\text{tw}(G - X) \leq 2^{15r+8r \log r}$, we obtain in this way a $(c_{\mathcal{F}} \cdot 40h^2 \cdot 2^{15r+8r \log r+5h \log h} \cdot k, 2^{15r+8r \log r+1} + h)$ -protrusion decomposition of G . We now consider again the encoder $\mathcal{E}_{\mathcal{FD}} = (\mathcal{C}_{\mathcal{FD}}, L_{\mathcal{C}_{\mathcal{FD}}})$ defined in Subsection 6.1, and by Corollary 1 we obtain a kernel of size at most

$$(1 + b(\mathcal{E}_{\mathcal{FD}}, g, 2^{15r+8r \log r+1} + h, \mathcal{G})) \cdot (c_{\mathcal{F}} \cdot 40h^2 \cdot 2^{15r+8r \log r+5h \log h} \cdot k), \quad (17)$$

where $b(\mathcal{E}_{\mathcal{FD}}, g, 2^{15r+8r \log r+1} + h, \mathcal{G})$ is the function defined in Lemma 3 and \mathcal{G} is the class of H -topological-minor-free graphs. \square

We finally present a deterministic kernel, whose drawback is that the running time is nonuniform on \mathcal{F} and H .

Theorem 11. *Let \mathcal{F} be a finite set of connected graphs containing at least one r -vertex planar graph F , let H be an h -vertex graph, and let $\text{CPFD}_{H\text{-top}}$ be the restriction of the CONNECTED-PLANAR- \mathcal{F} -DELETION problem to input graphs which exclude H as a topological minor. Then $\text{CPFD}_{H\text{-top}}$ admits a linear kernel of size at most $f(r, h) \cdot k$, where f is an explicit function depending only on r and h , defined in Equation (18) below.*

Proof: The main observation is that if $(G, k) \in \text{CPFD}_{H\text{-top}}$, then there exists a set $X \subseteq V(G)$ with $|X| \leq k$ such that $G - X$ is \mathcal{F} -minor-free. In particular, by Theorem 9 it holds that $\text{tw}(G - X) \leq 2^{15r+8r \log r}$. Therefore, we know by Theorem 2 that if $(G, k) \in \text{CPFD}_{H\text{-top}}$, then G admits a $(40 \cdot h^2 \cdot 2^{15r+8r \log r+5h \log h} \cdot k, 2^{15r+8r \log r+1} + h)$ -protrusion decomposition. Nevertheless, we do not have tools to efficiently find such linear decomposition. However, we use that, as observed in [7], a t -protrusion of size more than a prescribed number x in an n -vertex graph can be found in $n^{O(t)}$ steps, if it exists. Our kernelization algorithm proceeds as follows. We try to find a $(2^{15r+8r \log r+1} + h)$ -protrusion Y of size strictly larger than $x := b(\mathcal{E}_{\mathcal{FD}}, g, 2^{15r+8r \log r+1} + h, \mathcal{G})$, where $\mathcal{E}_{\mathcal{FD}}$ is the encoder for \mathcal{F} -DELETION described in Subsection 6.1, $b(\mathcal{E}_{\mathcal{FD}}, g, 2^{15r+8r \log r+1} + h, \mathcal{G})$ is the function defined in Lemma 3, and \mathcal{G} is the class of H -topological-minor-free graphs. If we succeed, we apply the protrusion replacement algorithm given by Theorem 1 and replace Y with another t -boundaried graph Y' such that $|Y'| \leq b(\mathcal{E}_{\mathcal{FD}}, g, 2^{15r+8r \log r+1} + h, \mathcal{G})$. The algorithm continues as far as we are able to find such large protrusion. At the end of this procedure, we either obtain an equivalent instance of size at most

$$b(\mathcal{E}_{\mathcal{FD}}, g, 2^{15r+8r \log r+1} + h, \mathcal{G}) \cdot 40 \cdot h^2 \cdot 2^{15r+8r \log r+5h \log h} \cdot k, \quad (18)$$

or otherwise we can correctly declare that (G, k) is a NO-instance. This kernelization algorithm runs in time $n^{O(2^{15r+8r \log r+1} + h)}$. \square

To conclude this section, we would like to note that the recent results of Chekuri and Chuzhoy [10] show that in Theorem 9, the inequality $\text{tw}(G) \leq 2^{15r+8r \log r}$ can be

replaced with $\text{tw}(G) = r^{O(1)}$. This directly implies that in Equations (17) and (18), as well as in the running time of the algorithm of Theorem 11, the term $2^{15r+8r \log r}$ can be replaced with $r^{O(1)}$. Nevertheless, we decided to keep the current bounds in order to be able to give explicit constants.

7 Further research

The methodology for performing explicit protrusion replacement via dynamic programming that we have presented is quite general, and it could also be used to obtain polynomial kernels (not necessarily linear). We have restricted ourselves to vertex-certifiable problems, but it seems plausible that our approach could be also extended to edge-certifiable problems or to problems on directed graphs.

We have presented in Section 6 a linear kernel for $\text{CONNECTED-PLANAR-}\mathcal{F}\text{-DELETION}$ when the input graph excludes a fixed graph H as a (topological) minor. The $\text{PLANAR-}\mathcal{F}\text{-DELETION}$ problem is known to admit a polynomial kernel on general graphs [17]. Nevertheless, this kernel has size $O(k^c)$, where c is a constant depending on \mathcal{F} that is upper-bounded by $2^{r^{10}}$, where r is the size of a largest graph in \mathcal{F} . The existence of a *uniform* polynomial kernel (that is, a polynomial kernel whose degree does not depend on the family \mathcal{F}) for $\text{PLANAR-}\mathcal{F}\text{-DELETION}$ on general graphs remains open.

As mentioned above, our linear kernel for $\text{PLANAR-}\mathcal{F}\text{-DELETION}$ requires that all graphs in the family \mathcal{F} are *connected*. It would be interesting to get rid of this assumption. On the other hand, in the linear kernel for $\text{CONNECTED-PLANAR-}\mathcal{F}\text{-DELETION}$ on H -topological-minor-free graphs given in Theorem 10, the randomization appears because we use the randomized constant-factor approximation for $\text{PLANAR-}\mathcal{F}\text{-DELETION}$ on general graphs [17], but for our kernel to be deterministic, it would be enough with a constant-factor approximation on H -topological-minor-free graphs, which is not known.

All the applications examined in this paper concerned parameterized problems tuned by a secondary parameter, i.e., r for the case of $r\text{-DOMINATING SET}$ and $r\text{-SCATTERED SET}$ and the size of the graphs in \mathcal{F} for the case of $\mathcal{F}\text{-DELETION}$. In all kernels derived for these problems, the dependency on this secondary parameter is triple-exponential, while the dependency on the choice of the excluded graph H is one exponent higher. Two questions arise:

- Extend our results to larger graph classes and more general problems. Also, improve the dependency of the size of the kernels on the “meta-parameters” associated with the problems (that is, r , \mathcal{F} , and H). Probably the recent results of Chekuri and Chuzhoy [10] can be used in this direction. Moreover, provide refinements of this framework that can lead to reasonable explicit bounds for the kernels for particular problems.
- Examine to what extent this exponential dependency is unavoidable under some assumptions based on automata theory or (parameterized) complexity theory. We suspect that the unification between dynamic programming and kernelization that we propose in this paper might offer a common understanding of the lower bounds in the running time of dynamic programming algorithms for certain problems (see [25, 26]) and the sizes of their corresponding kernels (see for instance [5, 6, 8, 12]). Finally, we refer the reader to [3] for constructibility issues of algebraic graph reduction.

Acknowledgement. We would like to thank the anonymous referees of the conference version of this paper for helpful remarks that improved the presentation of the manuscript.

References

1. I. Adler, F. Dorn, F. V. Fomin, I. Sau, and D. M. Thilikos. Faster parameterized algorithms for minor containment. *Theoretical Comput. Science*, 412(50):7018–7028, 2011.
2. J. Alber, M. Fellows, and R. Niedermeier. Polynomial-Time Data Reduction for Dominating Set. *Journal of the ACM*, 51(3):363–384, 2004.
3. S. Arnborg, B. Courcelle, A. Proskurowski, and D. Seese. An algebraic theory of graph reduction. *Journal of the ACM*, 40(5):1134–1164, 1993.
4. H. L. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM Journal on Computing*, 25(6):1305–1317, 1996.
5. H. L. Bodlaender. Kernelization: New Upper and Lower Bound Techniques. In *Proc. of the 4th International Workshop on Parameterized and Exact Computation (IWPEC)*, volume 5917 of *LNCS*, pages 17–37, 2009.
6. H. L. Bodlaender, R. G. Downey, M. R. Fellows, and D. Hermelin. On Problems without Polynomial Kernels (Extended Abstract). In *Proc. of the 35th International Colloquium on Automata, Languages and Programming (ICALP)*, pages 563–574, 2008.
7. H. L. Bodlaender, F. V. Fomin, D. Lokshtanov, E. Penninkx, S. Saurabh, and D. M. Thilikos. (Meta) Kernelization. In *Proc. of the 50th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 629–638. IEEE Computer Society, 2009.
8. H. L. Bodlaender, B. M. P. Jansen, and S. Kratsch. Cross-composition: A new technique for kernelization lower bounds. In *Proc. of the 28th International Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 9 of *LIPIcs*, pages 165–176, 2011.
9. H. L. Bodlaender and B. van Antwerpen-de Fluiter. Reduction algorithms for graphs of small treewidth. *Information and Computation*, 167(2):86–119, 2001.
10. C. Chekuri and J. Chuzhoy. Polynomial Bounds for the Grid-Minor Theorem. *CoRR*, abs/1305.6577, 2013.
11. B. Courcelle. The Monadic Second-Order Logic of Graphs. I. Recognizable Sets of Finite Graphs. *Information and Computation*, 85(1):12–75, 1990.
12. H. Dell and D. van Melkebeek. Satisfiability allows no nontrivial sparsification unless the polynomial-time hierarchy collapses. In *Proc. of the 42nd ACM symposium on Theory of computing (STOC)*, pages 251–260. ACM, 2010.
13. E. D. Demaine, F. V. Fomin, M. T. Hajiaghayi, and D. M. Thilikos. Fixed-parameter algorithms for (k, r) -center in planar graphs and map graphs. *ACM Transactions on Algorithms*, 1(1):33–47, 2005.
14. R. Diestel. *Graph Theory*, volume 173. Springer-Verlag, 2005.
15. R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer, 1999.
16. J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer Verlag, 2006.
17. F. V. Fomin, D. Lokshtanov, N. Misra, and S. Saurabh. Planar \mathcal{F} -Deletion: Approximation, Kernelization and Optimal FPT Algorithms. In *Proc. of the 53rd IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 470–479, 2012.
18. F. V. Fomin, D. Lokshtanov, S. Saurabh, and D. M. Thilikos. Bidimensionality and kernels. In *Proc. of the 21st ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 503–510. SIAM, 2010.
19. J. Guo and R. Niedermeier. Linear problem kernels for NP-hard problems on planar graphs. In *Proc. of the 34th International Colloquium on Automata, Languages and Programming (ICALP)*, volume 4596 of *LNCS*, pages 375–386, 2007.
20. K. ichi Kawarabayashi and Y. Kobayashi. Linear min-max relation between the treewidth of H -minor-free graphs and its largest grid. In *Proc. of the 29th International Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 14 of *LIPIcs*, pages 278–289, 2012.
21. K. ichi Kawarabayashi and B. Reed. A Separator Theorem in Minor-Closed Classes. In *Proc. of the 51st IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 153–162. IEEE Computer Society, 2010.
22. E. J. Kim, A. Langer, C. Paul, F. Reidl, P. Rossmanith, I. Sau, and S. Sikdar. Linear kernels and single-exponential algorithms via protrusion decompositions. In *Proc. of the 40th International Colloquium on Automata, Languages and Programming (ICALP)*, volume 7965 of *LNCS*, pages 613–624, 2013. Full version available at *CoRR*, abs/1207.0835.

- 23. T. Kloks. *Treewidth. Computations and Approximations*. Springer-Verlag, LNCS, 1994.
- 24. A. Leaf and P. D. Seymour. Treewidth and planar minors. Technical report, Princeton University, 2012.
- 25. D. Lokshtanov, D. Marx, and S. Saurabh. Known algorithms on graphs of bounded treewidth are probably optimal. In *Proc. of the 22nd ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 777–789. SIAM, 2011.
- 26. D. Lokshtanov, D. Marx, and S. Saurabh. Slightly superexponential parameterized problems. In *Proc. of the 22nd ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 760–776. SIAM, 2011.
- 27. N. Robertson and P. D. Seymour. Graph Minors. XIII. The Disjoint Paths Problem. *Journal of Combinatorial Theory, Series B*, 63(1):65–110, 1995.
- 28. B. van Antwerpen-de Fluiter. *Algorithms for graphs of small treewidth*. PhD thesis, Utrecht University, 1997.