Explicit Linear Kernels via Dynamic Programming*

Valentin Garnero[1], Christophe Paul[1], Ignasi Sau[1], and Dimitrios M. Thilikos[1,2]

1 **AlGCo project-team, CNRS and Université de Montpellier 2, LIRMM, Montpellier, France**
`FirstName.FamilyName@lirmm.fr`

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

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.

1998 ACM Subject Classification G.2.2 Graph Theory, F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases parameterized complexity, linear kernels, dynamic programming, protrusion replacement, graph minors

Digital Object Identifier 10.4230/LIPIcs.STACS.2014.312

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 [6, 7] 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 event linearly) on k. Finding

* 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".

31st Symposium on Theoretical Aspects of Computer Science (STACS'14).
Editors: Ernst W. Mayr and Natacha Portier; pp. 312–324

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

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 [11] to a 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.* [3] on graphs of bounded genus. After that, similar results have been obtained on larger sparse graph classes, such as graphs excluding a minor [9] or a topological minor [14].

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 [3], 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 [4] 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 13 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 [3, 8, 9, 14], 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 $r \geqslant 1$, 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) minor[1].

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. Due to space limitations, the proofs of the results marked with '[⋆]' can be found in the full version of this paper, which is permanently available at [10]. For the reader not familiar with the background used in previous work on this topic [3, 9, 14], some preliminaries can be found in [10], including graph minors, parameterized problems, (rooted) tree-decompositions, boundaried graphs, the canonical equivalence relation $\equiv_{\Pi,t}$ for a problem Π and an integer t, FII, protrusions, and protrusion decompositions. In Section 2 we introduce the basic definitions of our framework and present an explicit protrusion replacer. In Section 3 we show how to apply our methodology to various problems, and we conclude with some directions for further research in Section 4.

2 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 1** (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 $|S| \leqslant k$ (or $|S| \geqslant k$, 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 2.1 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 2.2 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 [10]). This refined equivalence relation allows us to provide an explicit upper bound on the size of its representatives (Lemma 11), as well as a linear-time algorithm to find them (Lemma 12). In Subsection 2.3 we use the previous ingredients to present an explicit

[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 1), for presenting it we should restate and extend many of the definitions and results given in Section 2 in order to deal with more general families of problems. Therefore, we decided not to include this family of problems in this article.

protrusion replacement rule (Theorem 13), 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.

2.1 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:

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

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 $\{v \in B \mid R(v) = \uparrow 1\}$. ◀

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

▶ **Definition 2** (Encoder). An *encoder* $\mathcal{E}$ is a pair $(\mathcal{C}, L_{\mathcal{C}})$ where

(i) $\mathcal{C}$ 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

(ii) $L_{\mathcal{C}}$ 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 $\mathcal{E}_{\mathrm{DS}} = (\mathcal{C}_{\mathrm{DS}}, L_{\mathcal{C}_{\mathrm{DS}}})$ 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 $\mathcal{C}_{\mathrm{DS}}$-encoding (that is, DP-table entry) $R \in \mathcal{C}_{\mathrm{DS}}(I)$, 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 $f_G^{\mathcal{E}_{\mathrm{DS}}} : \mathcal{C}_{\mathrm{DS}}(I) \to \mathbb{N} \cup \{+\infty\}$. Observe that if $B = \partial(G_B) = \emptyset$, then the value assigned to the encodings in $\mathcal{C}_{\mathrm{DS}}(\emptyset)$ is indeed the size of a minimum dominating set of G_B. ◇

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. (Maximization problems are treated similarly, see [10]

for the corresponding definitions of the functions $f_G^{\mathcal{E}}$ and $f_G^{\mathcal{E},g}$ defined below. The other definitions of this section remain unchanged.)

▶ **Definition 3** (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 $f_G^{\mathcal{E}} : \mathcal{C}(I) \to \mathbb{N} \cup \{+\infty\}$ as

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

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 3 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 $(G, S) \in L_\Pi$.

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 [3, 9, 14].

▶ **Definition 4** (Confined encoding)**.** An encoder $\mathcal{E}$ is *g-confined* if there exists a function $g : \mathbb{N} \to \mathbb{N}$ such that for any t-boundaried graph G with $\Lambda(G) = I$ it holds that either $\mathcal{C}_G^*(I) = \emptyset$ or

$$\max_{R \in \mathcal{C}_G^*(I)} f_G^{\mathcal{E}}(R) \;-\; \min_{R \in \mathcal{C}_G^*(I)} f_G^{\mathcal{E}}(R) \;\leqslant\; g(t). \tag{2}$$

See the figure in [10] 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 $f_G^{\mathcal{E}}(R)$. *Running example:* It is easy to observe that the encoder $\mathcal{E}_{\text{DS}}$ 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 $(G, S, R) \in L_{\mathcal{C}_{DS}}$. Then note that R_0 is satisfied by the set $S \cup \partial(G)$, so we have that for every encoding R, $f_G^{\mathcal{E}_{\text{DS}}}(R) + |\partial(G)| \geqslant f_G^{\mathcal{E}_{\text{DS}}}(R_0)$. It follows that $f_G^{\mathcal{E}_{\text{DS}}}(R_0) - \min_R f_G^{\mathcal{E}_{\text{DS}}}(R) \;\leqslant\; |\partial(G)| \leqslant t$, proving that the encoder is indeed g-confined. ◇

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 4, 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 [10]). To this end, given a function $g : \mathbb{N} \to \mathbb{N}$, we define the function $f_G^{\mathcal{E},g} : \mathcal{C}(I) \to \mathbb{N} \cup \{+\infty\}$ as

$$f_G^{\mathcal{E},g}(R) \;=\; \begin{cases} +\infty, & \text{if } f_G^{\mathcal{E}}(R) - g(t) > \min_{R \in \mathcal{C}(I)} f_G^{\mathcal{E}}(R) \\ f_G^{\mathcal{E}}(R), & \text{otherwise.} \end{cases} . \tag{3}$$

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.

2.2 Equivalence relations and representatives

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

▶ **Definition 5** (Equivalence relation $\sim_{\mathcal{E},g,t}$)**.** Let $\mathcal{E}$ be an encoder, let $g : \mathbb{N} \to \mathbb{N}$, 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

$$f_{G_1}^{\mathcal{E},g}(R) \;=\; f_{G_2}^{\mathcal{E},g}(R) + c\,. \tag{4}$$

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 (4) is unique, otherwise every integer c satisfies Equation (4). We define the following function $\Delta_{\mathcal{E},g,t} : \mathcal{F}_t \times \mathcal{F}_t \to \mathbb{Z}$, which is called, following the terminology from Bodlaender *et al.* [3], the *transposition function* for the equivalence relation $\sim_{\mathcal{E},g,t}$.

$$\Delta_{\mathcal{E},g,t}(G_1, G_2) \;=\; \begin{cases} c, & \text{if } G_1 \sim_{\mathcal{E},g,t} G_2 \text{ and Eq. (4) holds for a unique integer } c; \\ 0, & \text{if } G_1 \sim_{\mathcal{E},g,t} G_2 \text{ and Eq. (4) holds for every integer; and} \\ & \text{undefined otherwise.} \end{cases} \tag{5}$$

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 5.

▶ **Definition 6** (Equivalence relation $\sim_{\mathcal{E},g,t,\mathcal{G}}$)**.** Let $\mathcal{G}$ be a class of graphs and let $G_1, G_2 \in F_t \cap \mathcal{G}$.

(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 [6, 7]). 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 Section 3, where $\mathcal{G}$ will be a class of graphs that exclude some fixed graph as a (topological) minor[2], this will always be possible, and in this case it holds that $r_{\mathcal{G},t} \leqslant 2^{t \log t} \cdot h^t \cdot 2^{h^2}$.

For an encoder $\mathcal{E} = (\mathcal{C}, L_{\mathcal{C}})$, we let $s_{\mathcal{E}}(t) := \max_{I \subseteq \{1,\ldots,t\}} |\mathcal{C}(I)|$, 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 $r_{\mathcal{G},t}$.

[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 [3, 9, 14] what is used in the expressibility in CMSO logic of the *problems* defined on a graph class.

▶ **Lemma 7.** *Let $\mathcal{G}$ be a graph class whose membership can be expressed in MSO logic. For any encoder $\mathcal{E}$, any function $g : \mathbb{N} \to \mathbb{N}$, 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 $r(\mathcal{E}, g, t, \mathcal{G}) := (g(t) + 2)^{s\varepsilon(t)} \cdot 2^t \cdot r_{\mathcal{G},t}$.*

Proof. Let us first show that the equivalence relation $\sim_{\mathcal{E},g,t}$ has finite index. Indeed, let $I \subseteq \{1, \ldots, t\}$. 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 $\{1, \ldots, t\}$ is 2^t, we deduce that the overall number of equivalence classes of $\sim_{\mathcal{E},g,t}$ is at most $(g(t) + 2)^{s\varepsilon(t)} \cdot 2^t$. 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. ◀

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, the graph G_x can be naturally seen as a graph in $\mathcal{F}_t$. (A brief discussion can be found in [10], and we refer to [3] for more details.)

▶ **Definition 8** (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 8, 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 9.** [$\star$] *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 $\Delta_{\Pi,t}(G_1, G_2) = \Delta_{\mathcal{E},g,t}(G_1, G_2)$.*

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 10** (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 $\Delta_{\mathcal{E},g,t}(G, G') \leqslant 0$.

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 11.** [⋆] *Let $\mathcal{G}$ be a graph class whose membership can be expressed in MSO logic. For any encoder $\mathcal{E}$, any function $g : \mathbb{N} \to \mathbb{N}$, 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 7.*

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 [3, 9].

▶ **Lemma 12.** [⋆] *Let $\mathcal{G}$ be a graph class, let $\mathcal{E}$ be an encoder, let $g : \mathbb{N} \to \mathbb{N}$, 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 $\Delta_{\mathcal{E},g,t}(H, G) \leqslant 0$, where the constant in the "O" notation depends only on $\mathcal{E}, g, b, \mathcal{G}$, and t.*

2.3 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 13.** *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 $g : \mathbb{N} \to \mathbb{N}$, 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 $k' \leqslant k$ and Y' is a t-protrusion with $|Y'| \leqslant b(\mathcal{E}, g, t, \mathcal{G})$, where $b(\mathcal{E}, g, t, \mathcal{G})$ is the function defined in Lemma 11.*

Proof. By Lemma 7, the number of equivalence classes of the equivalence relation $\sim_{\mathcal{E},g,t,\mathcal{G}}$ is finite, and by Lemma 11 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 12 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 $\Delta_{\mathcal{E},g,t}(Y', Y) \leqslant 0$. Since $\mathcal{E}$ is a Π-encoder and $\sim_{\mathcal{E},g,t,\mathcal{G}}$ is DP-friendly, it follows from Lemma 9 that $Y \equiv_{\Pi,t} Y'$ and that

[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.

$\Delta_{\Pi,t}(Y',Y) = \Delta_{\mathcal{E},g,t}(Y',Y) \leqslant 0$. 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 $k' \leqslant k$ and $|Y'| \leqslant b(\mathcal{E},g,t,\mathcal{G})$. ◀

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 13 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 13 yields the following corollary.

▶ **Corollary 14.** *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 $g : \mathbb{N} \to \mathbb{N}$, 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 11.*

Proof. For $1 \leqslant i \leqslant \ell$, we apply the polynomial-time algorithm given by Theorem 13 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 $G' \in \mathcal{G}$, $k' \leqslant k$, and $|V(G')| \leqslant |Y_0| + \ell \cdot b(\mathcal{E},g,t,\mathcal{G}) \leqslant (1 + b(\mathcal{E},g,t,\mathcal{G}))\alpha \cdot k$. ◀

Notice that once we fix the problem Π and the class of graphs $\mathcal{G}$ where Corollary 14 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 11, and this, in turn, depends on the bound of Lemma 7. However, as we see in the next section, more moderate estimations can be extracted for particular families of parameterized problems.

3 Application to concrete problems

In this section we demonstrate the applicability of our framework by providing linear kernels for several problems on graphs excluding a fixed graph as a (topological) minor. Due to space limitations, we focus here on r-Dominating Set and Planar-$\mathcal{F}$-Deletion. The linear kernel for r-Scattered Set can be found in [10].

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 [14, Lemma 3, Proposition 1, and Theorem 1].

▶ **Theorem 15** (Kim *et al.* [14])**.** *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 $|X| \leqslant c \cdot k$ such that $\mathbf{tw}(G - X) \leqslant t$, 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 2.2, 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 $r_{\mathcal{G},t} \leqslant 2^{t \log t} \cdot h^t \cdot 2^{h^2}$. The details can be found in [10].

An explicit linear kernel for r-Dominating Set. Let $r \geqslant 1$ 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 $\lvert S\rvert \leqslant k$ 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.* [5], and it generalizes to one given for Dominating Set in the running example of Section 2. It can be found in [10], and we call it $\mathcal{E}_{r\mathrm{DS}} = (\mathcal{C}_{r\mathrm{DS}}, L_{\mathcal{C}_{r\mathrm{DS}}})$.

▶ **Lemma 16.** [⋆] *The encoder $\mathcal{E}_{r\mathrm{DS}}$ is a rDS-encoder. Furthermore, if $\mathcal{G}$ is an arbitrary class of graphs and $g(t) = t$, then the equivalence relation $\sim_{\mathcal{E}_{r\mathrm{DS}},g,t,\mathcal{G}}$ is DP-friendly.*

We now 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.* [9], but also uses the explicit combinatorial bound of Kawarabayashi and Kobayashi [12] on the relation between the treewidth and the largest grid minor on H-minor-free graphs, and the algorithmic results of Kawarabayashi and Reed [13] in order to obtain the claimed set X.

▶ **Theorem 17** (Fomin *et al.* [9])**.** *Let $r \geqslant 1$ be an integer, let H be an h-vertex apex graph, and let $r\mathrm{DS}_H$ be the restriction of the r-Dominating Set problem to input graphs which exclude H as a minor. If $(G, k) \in r\mathrm{DS}_H$, then there exists a set $X \subseteq V(G)$ such that $|X| = r \cdot 2^{O(h \log h)} \cdot k$ and $\mathbf{tw}(G - X) = r \cdot 2^{O(h \log h)}$. Moreover, given an instance (G, k) of $r\mathrm{DS}_H$ 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 18.** *Let $r \geqslant 1$ be an integer, let H be an h-vertex apex graph, and let $r\mathrm{DS}_H$ be the restriction of the r-Dominating Set problem to input graphs which exclude H as a minor. Then $r\mathrm{DS}_H$ 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 (6) below.*

Proof. Given an instance (G, k) of $r\mathrm{DS}_H$, we run the cubic algorithm given by Theorem 17 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 $\mathbf{tw}(G - X) = r \cdot 2^{O(h \log h)}$. In the latter case, we use the set X as input to the algorithm given by Theorem 15, 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 $\mathcal{E}_{r\mathrm{DS}} = (\mathcal{C}_{r\mathrm{DS}}, L_{\mathcal{C}_{r\mathrm{DS}}})$ defined in [10]. By Lemma 16, $\mathcal{E}_{r\mathrm{DS}}$ is an rDS-encoder and $\sim_{\mathcal{E}_{r\mathrm{DS}},g,t,\mathcal{G}}$ is DP-friendly, where $\mathcal{G}$ is the class of H-minor-free graphs and $g(t) = t$. It can be proved

that $s_{\mathcal{E}_{r\mathrm{DS}}}(t) \leqslant (2r+1)^t$ (see [10]). Therefore, we are in position to apply Corollary 14 and obtain a linear kernel for $r\mathrm{DS}_H$ of size at most

$$r^2 \cdot 2^{O(h \log h)} \cdot b\left(\mathcal{E}_{r\mathrm{DS}}, g, r \cdot 2^{O(h \log h)}, \mathcal{G}\right) \cdot k \ , \tag{6}$$

where $b\left(\mathcal{E}_{r\mathrm{DS}}, g, r \cdot 2^{O(h \log h)}, \mathcal{G}\right)$ is the function defined in Lemma 11. ◀

It can be easily checked that the multiplicative constant involved in the upper bound of Equation (6) 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.

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 $\lvert S\rvert \leqslant k$ 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. We obtain a linear kernel for the problem using two different approaches. The first one follows the same scheme as the one used so far, 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 15 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, that can be found in [10], 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.* [8] 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 of Theorem 13. 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 $\mathcal{F}$.

Our encoder for the $\mathcal{F}$-Deletion problem (see [10]) uses the dynamic programming machinery developed by Adler *et al.* [1]. 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 [3, 8], but for some families $\mathcal{F}$ containing disconnected graphs, $\mathcal{F}$-Deletion has not FII (see [14] for an example of such family).

▶ **Theorem 19.** [⋆] *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 $\mathrm{CP}\mathcal{F}\mathrm{D}_H$ be the restriction of the* Connected-Planar-$\mathcal{F}$-Deletion *problem to input graphs which exclude H as a minor. Then $\mathrm{CP}\mathcal{F}\mathrm{D}_H$ 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, which can be found in [10].*

4 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 is seems plausible that our approach could be also extended to edge-certifiable problems or to problems on directed graphs.

The linear kernel for Planar-$\mathcal{F}$-Deletion requires that all graphs in the family $\mathcal{F}$ are *connected*. It would be interesting to get rid of this assumption. All the applications examined in this paper concerned parameterized problems tuned by a secondary parameter, i.e., r for the case of r-Dominating Set and r-Scattered Set and the size of the graphs in $\mathcal{F}$ for the case of $\mathcal{F}$-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. Improving these dependencies on the "meta-parameters" is worth being investigated, as well as examining to what extent this exponential dependency is unavoidable under some assumptions based on automata theory or (parameterized) complexity theory.

References

1 Isolde Adler, Frederic Dorn, Fedor V. Fomin, Ignasi Sau, and Dimitrios M. Thilikos. Faster parameterized algorithms for minor containment. *Theoretical Comput. Science*, 412(50):7018–7028, 2011.

2 J. Alber, M.R. Fellows, and R. Niedermeier. Polynomial-Time Data Reduction for Dominating Set. *Journal of the ACM*, 51(3):363–384, 2004.

3 Hans L. Bodlaender, Fedor V. Fomin, Daniel Lokshtanov, Eelko Penninkx, Saket Saurabh, and Dimitrios M. Thilikos. (Meta) Kernelization. In *Proc. of the 50th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 629–638. IEEE Computer Society, 2009.

4 Bruno Courcelle. The Monadic Second-Order Logic of Graphs. I. Recognizable Sets of Finite Graphs. *Information and Computation*, 85(1):12–75, 1990.

5 Erik D. Demaine, Fedor V. Fomin, Mohammad Taghi Hajiaghayi, and Dimitrios M. Thilikos. Fixed-parameter algorithms for (k, r)-center in planar graphs and map graphs. *ACM Transactions on Algorithms*, 1(1):33–47, 2005.

6 R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer, 1999.

7 Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Springer Verlag, 2006.

8 Fedor V. Fomin, Daniel Lokshtanov, Neeldhara Misra, and Saket 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.

9 Fedor V. Fomin, Daniel Lokshtanov, Saket Saurabh, and Dimitrios M. Thilikos. Bidimensionality and kernels. In *Proc. of the 21st ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 503–510. SIAM, 2010.

10 Valentin Garnero, Christophe Paul, Ignasi Sau, and Dimitrios M. Thilikos. Explicit linear lernels via dynamic programming. *CoRR*, abs/1312.6585, 2013.

11 Jiong Guo and Rolf 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.

12 Ken ichi Kawarabayashi and Yusuke 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.

13 Ken ichi Kawarabayashi and Bruce 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.

14 Eun Jung Kim, Alexander Langer, Christophe Paul, Felix Reidl, Peter Rossmanith, Ignasi Sau, and Somnath 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.