A Retrospective on (Meta) Kernelization

Dimitrios M. Thilikos

A Retrospective on (Meta) Kernelization

Dimitrios M. Thilikos[1][0000−0003−0470−1800]

November 18, 2020

«Een mens zijn zin
is een mens zijn leven.»

Abstract. In parameterized complexity, a *kernelization algorithm* can be seen as a reduction of a parameterized problem to itself, so that the produced equivalent instance has size depending exclusively on the parameter. If this size is polynomial, then then we say that the parameterized problem in question admits a polynomial kernelization algorithm. Kernelization can be seen as a formalization of the notion of preprocessing and has occupied a big part of the research on Multi-variate Algorithmics. The first algorithmic meta-theorem on kernelization appeared in [*H. L. Bodlaender, F. V. Fomin, D. Lokshtanov, E. Penninkx, S. Saurabh, and D. M. Thilikos. (Meta) kernelization. J. ACM, 63(5):44:1–44:69, 2016*] and unified a large family of previously known kernelization results on problems defined on topologically embeddable graphs. In this exposition we present the central results of this paper. During our presentation we pay attention to the abstractions on which the results where founded and take into account the subsequent advancements on this topic.

Keywords: Parameterized problems · Parameterized Algorithms · Kernelization Algorithms · Algorithmic Meta-theorems Finite Index · Finite Integer Index · Monadic Second Order Logic · Treewidth · Protrusion Decompositions, Bidimensionality · Separability.

1 Introduction

Parametrized complexity, introduced by Downey and Fellows in early 90's [1, 36–39], has nowadays evolved to a mature field of Theoretical Computer Science (see [25, 35, 40, 45, 83] for related textbooks). The key idea is to treat computational problems as *bivariate entities* where, apart from the size of the input, the

[1]Supported by projects DEMOGRAPH (ANR-16-CE40-0028) and ESIGMA (ANR-17-CE23-0010) and by the Research Council of Norway and the French Ministry of Europe and Foreign Affairs, via the Franco-Norwegian project PHC AURORA 2019.

algorithm complexity is evaluated with respect to some *parameter* that quantifies structural properties of the input. This approach is justified by the fact that, for many computational problems, the inherent combinatorial explosion of their complexity depends exclusively by the parameter, whose value is expected to be small in certain applications. In this framework we deal with parameterized problems whose instances are pairs $(x, k) \in \Sigma^* \times \mathbb{N}$ and we look for algorithms whose running time is exponential in the parameter k but polynomial in the input size $n = |x|$. In other words, we are aiming for algorithms that run in time $f(k) \cdot n^c$ where f is a function depending only on the parameter and where c is a constant. The parameterized problems for which such algorithms exist constitute the parameterized complexity class FPT. In their foundational work [1, 36–39], Downey and Fellows invented a series of parameterized complexity classes and proposed special types of reductions such that hardness for some of the above classes gives plausible evidence that a problem does not belong to FPT.

Kernelization is a prominent field of parameterized complexity that has rapidly developed during the last two decades (see [57] for a recent textbook). A *kernelization algorithm* (or simply a *kernelization*) is a polynomial time algorithm that transforms every instance (x, k) of a parameterized problem to an equivalent one (x', k'), whose size depends exclusively on the parameter. We refer to the size of (x', k'), measured as a function of k, as the *size* of the kernelization. Kernelization algorithms can be seen as preprocessing procedures with performance guarantee. For this reason, kernelization has been considered as an attempt to mathematically formalize the concept of preprocessing. Clearly, the size of a kernelization is important as it evaluates how good is the preprocessing that it achieves.

It is known that every every decidable parameterized problem parameterized problem in FPT admits a kernelization and vice versa. However, it is not always the case that a problem in FPT admits a *polynomial size* kernelization. The running challenge is to distinguish which problems in FPT admit polynomial size kernelizations and, if this is the case, to optimize the corresponding size function. In this direction, diverse algorithmic techniques have been invented (see [41, 57]) while, on the negative side, new complexity theory tools have been introduced for proving that polynomial size kernelization would imply some unexpected collapse in classic computational complexity [11, 27] (see also [57]).

Algorithmic meta-theorems can be seen as theoretical results providing general conditions that, when satisfied, automatically imply the existence of efficient algorithms for wide families of problems. The term "Algorithmic Meta-Theorem" was introduced by Martin Grohe in his seminal exposition in [65] (see also [79] and [67]). Typically the conditions of such results have a logical part, concerning the descriptive complexity of the problem, and a combinatorial part that concerns the inputs of the problem. Algorithmic meta-theorems reveal interesting relations between logic and combinatorial structures and yield a better understanding of the nature of general algorithmic techniques.

Many known algorithmic meta-theorems have been stated using the multivariate framework of *parameterized complexity*. This is due to the fact that the

concept of problem parameterization may serve as a way to formalize the imposed structural restrictions. The most classic example of an Algorithmic Meta-theorem leading to a massive classification of parameterized problems in FPT is the celebrated Courcelle's theorem [21, 23]: *Checking graph properties definable in Counting Monadic Second Order Logic (CMSO), when parameterized by the treewidth of the input graph, belongs in* FPT (see subsection 2.3 for the formal definition of treewidth). For other algorithmic meta-theorems permitting the classification of parameterized problems in FPT and containing different trade-offs between the logical part and the combinatorial part, see [26, 58, 81].

The purpose of this exposition is to present the main algorithmic meta-theorems on kernelization for problems on graphs. The first results of this kind appeared in [14] and they essentially initiated the research on meta-algorithmic for kernelization (see also [8] for an earlier version). The results in [14] subsumed several previous results on kernelization for problems on planar graphs such as [3, 4, 15, 16, 18, 19, 62, 69, 70, 75, 80, 82]. Moreover, the algorithmic techniques in [14] introduced new concepts and tools that were of use in later investigations [42, 54, 55, 59–61, 74, 76, 77, 87]. Subsequently, several results appeared in [54, 55] that extended the original theorems from [14] or applied the ideas of [14] in different combinatorial settings (see e.g., [56, 59, 61, 63, 74, 77, 87]).

We provide a description of the main ideas of [14], taking into account all the improvements and generalizations known up to now [49, 50, 54, 55, 76]. We present two main algorithmic meta-theorems on kernels for problems on graphs that can be seen as abstract versions of the results in [14]. We also give sketches of some of the proofs in order to expose the core ideas, as they were originally conceived in [14]. Our exposition is self-contained and pays special attention to "mathematical formalism" and "details". In fact we provide an alternative to the formalism of [14] (mostly inspired by [54, 55]) that is versatile enough so to form the base of further investigations on this topic.

2 Definitions

We use $\mathbb{Z}$ for the set of integers and $\mathbb{N}$ for the set of non-negative integers. Given some $\ell \in \mathbb{N}$, we denote $[\ell] = \{1, \ldots, \ell\}$. We also use $\mathbb{R}_{>0}$ for the set of all positive reals. Consider a tuple $\mathbf{t} = (x_1, \ldots, x_\ell) \in \mathbb{N}^\ell$ and two functions $\chi, \psi : \mathbb{N} \to \mathbb{N}$. Given a function $f : A \to B$, we also assume its set-extension $f : 2^A \to 2^B$ so that for every $X \in 2^A$, it holds $f(X) = \bigcup_{x \in X} f(x)$.

We write $\chi(n) = O_{\mathbf{t}}(\psi(n))$ in order to denote that there exists a function $\phi : \mathbb{N}^\ell \to \mathbb{N}$ such that $\chi(n) = O(\phi(\mathbf{t}) \cdot \psi(n))$. Also, given a function $f : \mathbb{N} \to \mathbb{N}$ and a tuple $\mathbf{t} = (x_1, \ldots, x_\ell) \in \mathbb{N}^\ell$, we write $\chi(n) = O_{f,\mathbf{t}}(\psi(n))$ in order to denote that there exists a function $\phi : \mathbb{N} \to \mathbb{N}$ such that $\chi(n) = O(\phi(f(\mathbf{t})) \cdot \psi(n))$.

All graphs that we consider are simple, finite, and undirected. Given a graph G, we denote by $V(G)$ and $E(G)$ the set of its vertices and edges respectively. The *size* of a graph G is the number of its vertices and is denoted by $|G|$.

2.1 Parameterized problems on graphs.

We start by giving the definitions of a series of algorithmic concepts.

A *parameterized problem* is any subset Π of the set $\Sigma^* \times \mathbb{N}$ where Σ is some alphabet with at least two symbols. If $(x,k) \in \Sigma^* \times \mathbb{N}$ belongs in Π, then we say that (x,k) is a yes-*instance* of Π, otherwise we say that (x,k) is a no-*instance* of Π. Given two parameterized problems $\Pi, \Pi' \subseteq \Sigma^* \times \mathbb{N}$ and two instances $(x,k),(x',k') \in \Sigma^* \times \mathbb{N}$, we say that (x,k) and (x',k') are *equivalent instances* (*of Π and Π'*) if $(x,k) \in \Pi \iff (x',k') \in \Pi'$.

An *annotated graph* is a pair (G,S) where G is a graph and $S \subseteq V(G)$. We deal exclusively with parameterized problems either on graphs or on annotated graphs, therefore we see them as subsets of $\mathcal{G}_{\text{all}} \times \mathbb{N}$ or subsets of $\mathcal{A}_{\text{all}} \times \mathbb{N}$ where $\mathcal{G}_{\text{all}}$ is the set of all graphs and $\mathcal{A}_{\text{all}}$ the set of all annotated graphs. From now on, whenever we present some algorithm on some problem on (annotated) graphs we evaluate its time complexity in terms of the size of the graph in their input that we always denote by n.

A subset of $\mathcal{A}_{\text{all}}$ is called *vertex-subset property*.

Given a parameterized problem on graphs $\Pi \subseteq \mathcal{G}_{\text{all}} \times \mathbb{N}$ and a graph class $\mathcal{G} \subseteq \mathcal{G}_{\text{all}}$, we define *the restriction of Π in $\mathcal{G}$* by

$$\Pi \Cap \mathcal{G} = \{(G,k) \mid (G,k) \in \Pi \text{ and } G \in \mathcal{G}\}.$$

Annotated graphs. Given a vertex-subset property $\mathcal{A} \subseteq \mathcal{A}_{\text{all}}$ and some graph class $\mathcal{G} \subseteq \mathcal{G}_{\text{all}}$, we define

$$\mathcal{A} \Cap \mathcal{G} = \{(G,S) \mid (G,S) \in \mathcal{A} \wedge G \in \mathcal{G}\}$$

and we call $\mathcal{A} \Cap \mathcal{G}$ the *restriction* of $\mathcal{A}$ to $\mathcal{G}$. We define the *domain* of $\mathcal{A}$ as

$$\mathsf{dom}(\mathcal{A}) = \{G \mid \exists k \in \mathbb{N} : (G,k) \in \mathcal{A}\}.$$

Notice that $\mathsf{dom}(\mathcal{A} \Cap \mathcal{G}) \subseteq \mathcal{G}$.

Problem defining pairs and vertex-subset problems. A pair $(\mathcal{A}, \mathsf{opt})$ where $\mathcal{A} \subseteq \mathcal{A}_{\text{all}}$ and $\mathsf{opt} \in \{\min, \max\}$ is called *problem defining pair* or, in short, *defining pair*. Given a defining pair $(\mathcal{A}, \mathsf{opt})$, we define the corresponding *vertex-subset problem* as the parameterized problem on graphs

$$\Pi_{\mathcal{A},\mathsf{opt}} = \{(G,k) \mid \exists S \subseteq V(G) : |S| \gtrless k \ \wedge \ (G,S) \in \mathcal{A}\}$$

where "$\gtrless$" is interpreted as "$\leq$", in case $\mathsf{opt} = \min$ and as "$\geq$", in case $\mathsf{opt} = \max$.

A *vertex-subset problem* is any parameterized problem on graphs Π such that $\Pi = \Pi_{\mathcal{A},\mathsf{opt}}$, for some defining pair $(\mathcal{A}, \mathsf{opt})$. In particular, we can see a vertex-subset problem as a problem that is generated by some defining pair $(\mathcal{A}, \mathsf{opt})$.

We also define the *annotated version* of $\Pi_{\mathcal{A},\mathsf{opt}}$ as follows. If $\mathsf{opt} = \min$, then

$$\Pi^{\alpha}_{\mathcal{A},\min} = \{((G,Y),k) \mid \exists S \subseteq Y : |S| \leq k \ \wedge \ (G,S) \in \mathcal{A}\},$$

while if $\mathsf{opt} = \max$, then

$$\Pi^{\alpha}_{\mathcal{A},\max} = \{((G,Y),k) \mid \exists S \subseteq V(G) : |S \cap Y| \geq k \ \wedge \ (G,S) \in \mathcal{A}\}.$$

Clearly, $\Pi_{\mathcal{A},\mathsf{opt}}$ can be seen as a special case of $\Pi^{\alpha}_{\mathcal{A},\mathsf{opt}}$ if we consider only the pairs whose annotated graph is of the form $(G, V(G))$.

Graph parameters. A *graph parameter* is any partial function mapping graphs to non-negative integers. Given a defining pair $(\mathcal{A}, \mathsf{opt})$, we define the associated graph parameter $\mathbf{p}_{\mathcal{A},\mathsf{opt}} : \mathcal{G}_{\mathrm{all}} \to \mathbb{N}$ so that

$$\mathbf{p}_{\mathcal{A},\mathsf{opt}}(G) = \mathsf{opt}\,\{k \mid (G,k) \in \Pi_{\mathcal{A},\mathsf{opt}}\},$$

while, if $G \notin \mathsf{dom}(\mathcal{A})$, then $\mathbf{p}_{\mathcal{A},\mathsf{opt}}(G)$ is undefined. Notice that $\mathsf{dom}(\mathcal{A})$ is the domain of the graph parameter $\mathbf{p}_{\mathcal{A},\mathsf{opt}}$. Also we define the function $\mathsf{sol}_{\mathcal{A},\mathsf{opt}}$ such that for every $G \in \mathcal{G}_{\mathrm{all}}$,

$$\mathsf{sol}_{\mathcal{A},\mathsf{opt}}(G) = \{S \mid (G,S) \in \mathcal{A} \wedge |S| = \mathbf{p}_{\mathcal{A},\mathsf{opt}}(G)\}.$$

Clearly $\mathsf{sol}_{\mathcal{A},\mathsf{opt}}(G) \neq \emptyset \iff G \in \mathsf{dom}(\mathcal{A})$. We can see $\mathsf{sol}_{\mathcal{A},\mathsf{opt}}(G)$ as the set of all *optimal solutions* for the vertex-subset problem $\Pi_{\mathcal{A},\mathsf{opt}}$.

2.2 (Bi)kernelization

Kernelization can be seen as a polynomial reduction of a parameterized problem to itself. Here we present it as a special case of a more general concept of reduction called *bikernel* (see [60, 71, 72] for earlier uses of this concept).

Let $\Pi_1, \Pi_2 \subseteq \Sigma^* \times \mathbb{N}$ be two parameterized problems. A *bikernelization algorithm* (or simply *a bikernelization*) from Π_1 to Π_2 is a polynomial-time computable function $\mathsf{A} : \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N}$ such that

1. $\forall (x,k) \in \Sigma^* \times \mathbb{N}$ $(x,k) \in \Pi_1 \iff \mathsf{A}(x,k) \in \Pi_2$ (i.e., (x,k) and $\mathsf{A}(x,k)$ are equivalent instances of Π_1 and Π_2 respectively) and
2. there exists a computable function $s : \mathbb{N} \to \mathbb{N}$ such that, for every pair $(x,k) \in \Sigma^* \times \mathbb{N}$, it holds that if $\mathsf{A}(x,k) = (x',k')$, then $\max\{|x'|, k'\} \leq s(k)$.

Given a parameterized problem $\Pi \subseteq \Sigma^* \times \mathbb{N}$, we define a *kernelization algorithm* (or simply *a kernelization*) for Π as a bikernelization algorithm from Π to Π.

We call the function s above *the size* of the (bi-)kernelization A. If s is a polynomial (resp. linear) function, we say that A is a *polynomial-size (resp. linear-size) (bi-)kernelization from Π_1 to Π_2*. Also we use the term *time of a (bi-)kernelization* in order to refer to the time of the (bi-)kernelization algorithm A. We stress that when x encodes a graph G, then $|x|$ is the encoding size of G.

It is known that every decidable parameterized problem that is in FPT admits a kernelization and vice versa. However, it is not always the case that there is a kernelization of polynomial size. As we already mentioned in the introduction, a central question of parameterized complexity is to distinguish which problems in FPT have kernelizations of polynomial size and which do not.

2.3 Graph decompositions

We now provide several combinatorial concepts that will be useful for our proofs, namely, tree decompositions, protrusion decompositions, and the notion of protrusion decomposability.

Tree decompositions. Let G be a graph. A *tree decomposition* of G is a pair $D = (T, \chi)$, where T is a tree and $\chi : V(T) \to 2^{V(G)}$ such that:

1. $\bigcup_{q \in V(T)} \chi(q) = V(G)$,
2. for every edge $\{u, v\} \in E$, there is a $q \in V(T)$ such that $\{u, v\} \subseteq \chi(q)$, and
3. for each $v \in V(G)$ the set $\{t \mid v \in \chi(t)\}$ induces a connected subgraph of T.

We call the vertices of T *nodes* of D and the images of χ *bags* of D. The *width* of a tree decomposition $D = (T, \chi)$ is $\max\{|\chi(q)| \mid q \in V(T)\} - 1$. The treewidth of a G is the minimum width over all tree decompositions of G.

Protrusion decompositions. We denote by $\partial_G(S)$ the vertices of S that have neighbors outside S and by $N_G(S)$ the neighbours of vertices in S that do not belong to S. We also st $N_G[S] = S \cup N_G(S)$. Let G be a graph and let $R \subseteq V(G)$. We say that R is a β-*protrusion* of G if $\max\{|\partial_G(R)|, \mathbf{tw}(G[R])\} \leq \beta$. An (α, β)-*protrusion-decomposition* of a graph G is a partition $\mathcal{P} = \{X_0, X_1, \ldots, X_\ell\}$ of $V(G)$ such that

1. $\max\{\ell, |X_0|\} \leq \alpha$,
2. for every $i \in [\ell]$, the set $R_i = N_G[X_i]$ is a β-protrusion of G and
3. for every $i \in [\ell]$, $N_G(X_i) \subseteq X_0$.

We call the set X_0 the *core* of the (α, β)-*protrusion-decomposition* $\mathcal{P}$. We also call the sets $X_i, i \in [\ell]$ *flaps* of $\mathcal{P}$. Intuitively, an (α, β)-protrusion-decomposition of a graph can be seen as a partition into at most $\alpha + 1$ sets where one of them, the core, has size at most α and each of the rest, the flaps, has treewidth at most β and has at most β neighbors, all contained in the core.

Protrusion decompositions served as the main combinatorial tool of [14].

Protrusion-decomposability. Given a defining pair $(\mathcal{A}, \mathsf{opt})$, a function $\mathsf{T} : \mathbb{N} \to \mathbb{N}$, and a real constant $c > 0$, we say that the defining pair $(\mathcal{A}, \mathsf{opt})$ is *c-protrusion-decomposable in time* $\mathsf{T}(n)$, if there exists an algorithm that given a $(G, k) \in \mathsf{dom}(\mathcal{A}) \times \mathbb{N}$, either outputs that $\mathbf{p}_{\mathcal{A},\mathsf{opt}}(G) > k$ or outputs a $(c \cdot k, c)$-protrusion-decomposition of G in time $T(n)$ (recall that by n we always denote the size of the input graph G). It is important to observe that in the specifications of this algorithm we *demand* that its input should contain a graph in the domain of $\mathcal{A}$.

2.4 Boundaried graphs

Let $t \in \mathbb{N}$. A *t-boundaried graph* is a triple $\mathbf{G} = (G, B, \rho)$ where G is a graph, $B \subseteq V(G)$, $|B| = t$, and $\rho : B \to [t]$ is a bijection. We call G *underlying* graph of $\mathbf{G}$, B *the boundary* of $\mathbf{G}$, and we define the *size* of $\mathbf{G}$, denoted by $|\mathbf{G}|$, as the

size of G. We say that $\mathbf{G}_1 = (G_1, B_1, \rho_1)$ and $\mathbf{G}_2 = (G_2, B_2, \rho_2)$ are *isomorphic* if there is an isomorphism from G_1 to G_2 that extends the bijection $\rho_2^{-1} \circ \rho_1$.

We call *t-boundaried annotated graph* the pair $(\mathbf{G}, S)$, where $\mathbf{G} = (G, B, \rho)$ and $S \subseteq V(G)$. We say that two t-boundaried annotated graphs $(\mathbf{G}_1, S_1)$ and $(\mathbf{G}_2, S_2)$ (where $\mathbf{G}_1 = (G_1, B_1, \rho_1)$ and $\mathbf{G}_2 = (G_2, B_2, \rho_2)$) are *compatible* if $\rho_2^{-1} \circ \rho_1$ is an isomorphism from $G_1[B_1]$ to $G_2[B_2]$ and $\rho(B_1 \cap S_1) = \rho(B_2 \cap S_2)$. The *size* of $(\mathbf{G}, S)$ is defined to be the size of $\mathbf{G}$.

Given two compatible t-boundaried annotated graphs $(\mathbf{G}_1, S_1)$ and $(\mathbf{G}_2, S_2)$ where $\mathbf{G}_1 = (G_1, B_1, \rho_1)$ and $\mathbf{G}_2 = (G_2, B_2, \rho_2)$, we define $\mathbf{G}_1 \oplus \mathbf{G}_2$ as the annotated graph (G, S) where G is obtained if we take the disjoint union of G_1 and G_2 and, for every $i \in [|B_1|]$, we identify vertices $\rho_1^{-1}(i)$ and $\rho_2^{-1}(i)$ and S is obtained if we take the union of S_1 and S_2 and, for each $i \in \rho(B_1 \cap S_1)$, we identify the vertex $\rho_1^{-1}(i)$ with the vertex $\rho_2^{-1}(i)$. We agree that, during vertex identifications, the vertices of B_1 prevail those of B_2.

3 Meta-theorems on kernels

3.1 Meta-kernels for subset properties with finite index

Finite index. Let $\mathcal{A}$ be a vertex-subset property and $t \in \mathbb{N}$. Let also $(\mathbf{G}_1, S_1)$ and $(\mathbf{G}_2, S_2)$ be two compatible t-boundaried annotated graphs. We say that $(\mathbf{G}_1, S_1) \equiv_{\mathcal{A},t} (\mathbf{G}_2, S_2)$ if for every t-boundaried annotated graph $(\mathbf{F}, S_F)$ that is compatible with $(\mathbf{G}_1, S_1)$, it holds that

$$(\mathbf{F}, S_F) \oplus (\mathbf{G}_1, S_1) \in \mathcal{A} \iff (\mathbf{F}, S_F) \oplus (\mathbf{G}_2, S_2) \in \mathcal{A}.$$

It is easy to observe that $\equiv_{\mathcal{A},t}$ is an equivalence relation. For every $t \in \mathbb{N}$, we set up a set of representatives $\mathsf{rep}_{\mathcal{A}}(t)$ containing a minimum size t-boundaried annotated graph from each equivalence class of $\equiv_{\mathcal{A},t}$.

Given an increasing function $f : \mathbb{N} \to \mathbb{N}$, we say that a subset property $\mathcal{A}$ belongs in $\mathsf{FI}(f)$ if for every $t \in \mathbb{N}$, $f(t)$ is an upper bound to the size of all the t-boundaried annotated graphs in $\mathsf{rep}_{\mathcal{A}}(t)$. We say that a subset property $\mathcal{A}$ has *finite index* if it belongs in $\mathsf{FI}(f)$ for some increasing function $f : \mathbb{N} \to \mathbb{N}$. Notice that $\mathcal{A}$ has finite index if and only if $\equiv_{\mathcal{A},t}$ has a finite number of equivalence classes, for every $t \in \mathbb{N}$. The notion of finite index was central in the proof of Courcelle's theorem in [21], see [6, 9, 13, 17, 20, 22, 24] for related bibliography.

Our first result, corresponding to Theorem 1.1 of [14], is stated in terms of bikernelization algorithms.

Theorem 1. *Let $f : \mathbb{N} \to \mathbb{N}$ be an increasing function, $\mathcal{A} \in \mathsf{FI}(f)$, $\mathsf{opt} \in \{\min, \max\}$, and $c \in \mathbb{R}_{>0}$. If*

1. *$\mathsf{dom}(\mathcal{A})$ is computable in polynomial time $\mathsf{T}^1(n)$,*
2. *$(\mathcal{A}, \mathsf{opt})$ is c-protrusion-decomposable in polynomial time $\mathsf{T}^2(n)$,*

then there is a polynomial size bikernelization from $\Pi_{\mathcal{A},\mathsf{opt}}$ to $\Pi^{\alpha}_{\mathcal{A},\mathsf{opt}}$, of size $O(k^2)$, running in polynomial time $\mathsf{T}^1(n) + \mathsf{T}^2(n) + O_{f,c}(n)$.

The main meta-algorithmic consequence of Theorem 1, corresponding to Theorem 1.2 of [14], is the following.

Theorem 2. *Let $f : \mathbb{N} \to \mathbb{N}$ be an increasing function, $\mathcal{A} \in \mathsf{FI}(f)$, $\mathsf{opt} \in \{\min, \max\}$, and $c \in \mathbb{R}_{>0}$. If*

1. *$\mathsf{dom}(\mathcal{A})$ is computable in polynomial time $\mathsf{T}^1(n)$,*
2. *$(\mathcal{A}, \mathsf{opt})$ is c-protrusion-decomposable in polynomial time $\mathsf{T}^2(n)$,*
3. *$\Pi_{\mathcal{A},\mathsf{opt}}$ is NP-hard, and*
4. *$\Pi^{\alpha}_{\mathcal{A},\mathsf{opt}} \in \mathsf{NP}$,*

then $\Pi_{\mathcal{A},\mathsf{opt}}$ admits a polynomial size kernelization of size $k^{O_{f,c}(1)}$, running in polynomial time $\mathsf{T}^1(n) + \mathsf{T}^2(n) + O_{f,c}(n) + k^{O_{f,c}(1)}$.

In what follows in this subsection, we give the main steps of the proofs of Theorem 1 and Theorem 2. A useful ingredient is the following:

Lemma 1. *Let $f : \mathbb{N} \to \mathbb{N}$ be an increasing function, $\mathcal{A} \in \mathsf{FI}(f)$, and $\mathsf{opt} \in \{\min, \max\}$. For every t there exists an algorithm that, given two compatible t-boundaried annotated graphs $(\mathbf{G}_1, S_1)$ and $(\mathbf{G}_2, S_2)$, outputs a set $R \subseteq V(G_1)$ such that*

- *$(\mathbf{G}_1, R)$ and $(\mathbf{G}_1, S_1)$ are compatible,*
- *$(\mathbf{G}_2, S_2) \oplus (\mathbf{G}_1, R) \in \mathcal{A}$,*
- *in case $\mathsf{opt} = min$, then $R \subseteq S_1$ and $|R|$ is minimized, and*
- *in case $\mathsf{opt} = max$, then $|S_1 \cap R|$ is maximized,*

or reports that such a set R does not exist. Moreover, given that the underlying graph of $(\mathbf{G}_1, S_1) \oplus (\mathbf{G}_2, S_2)$ is G, this algorithm runs in time $O_{f,\mathbf{tw}(G),t}(|G|)$.

The proof of Lemma 1 is based on the fact that there is an algorithm that, given a vertex-subset property $\mathcal{A} \in \mathsf{FI}(f)$ (for some increasing function $f : \mathbb{N} \to \mathbb{N}$) and a graph G, outputs, if exists, a minimum or maximum (depending on the value of $\mathsf{opt} \in \{\max, \min\}$) size set S where $(G, S) \in \mathcal{A}$, in time $O_{f,\mathbf{tw}(G)}(|G|)$. This fact follows by making use of the results from [17, Theorem 5] for finite index vertex-subset properties, see also [6] and [14, Lemma 5.2].

The first important step towards proving Theorem 1 is the following.

Lemma 2. *Let $f : \mathbb{N} \to \mathbb{N}$ be an increasing function, $\mathcal{A} \in \mathsf{FI}(f)$, $\mathsf{opt} \in \{\max, \min\}$, and $t \in \mathbb{N}$. Then there exists an algorithm that, given an instance $((G, Y), k)$ of $\Pi^{\alpha}_{\mathcal{A},\mathsf{opt}}$ and a t-protrusion X of G, outputs an equivalent instance $((G, Y'), k)$ of $\Pi^{\alpha}_{\mathcal{A},\mathsf{opt}}$, where $|Y' \cap X| \leq O_{f,t}(k)$ and $Y' \subseteq Y$, in time $O_{f,t}(|X|)$.*

Proof (Proof (sketch).). We present the proof only for the case where $\mathsf{opt} = \min$. The case where $\mathsf{opt} = \max$ is similar (but not the same, see [14, Lemma 5.14]).

Let $B = \partial_G(X)$ and $t' = |B| \leq t$. We consider the t'-boundaried annotated graph $(\mathbf{X}, Z)$ where $\mathbf{X} = (G[X], B, \rho)$ (ρ is some, arbitrarily chosen, bijection

from $\partial_G(X)$ to $[t']$) and $Z = Y \cap X$. Recall that $\mathbf{tw}(G_X) \leq t$. We also consider the t'-boundaried annotated graph $(\mathbf{F}, R)$ where $\mathbf{F} = (G \setminus (X \setminus B), B, \rho)$ and $R = Y \setminus (X \setminus B)$. Clearly, $(G, Y) = (\mathbf{F}, R) \oplus (\mathbf{X}, Z)$.

Let now $\mathfrak{L} = (\mathbf{H}, W) \in \mathsf{rep}_{\mathcal{A}}(t')$. We apply the algorithm of Lemma 1 on $(\mathbf{H}, W)$ and $(\mathbf{X}, Z)$ and find, if exists, a minimum size set $S_{\mathfrak{L}} \subseteq Z$ such that $(\mathbf{X}, Z)$ and $(\mathbf{X}, S_{\mathfrak{L}})$ are compatible and $(\mathbf{H}, W) \oplus (\mathbf{X}, S_{\mathfrak{L}}) \in \mathcal{A}$. If such an $S_{\mathfrak{L}}$ does not exist or it has more than k vertices, then we set $S_{\mathfrak{L}} = \emptyset$. We define $Y' = R \cup W$ where

$$W = \bigcup_{\mathfrak{L} \in \mathsf{rep}_{\mathcal{A}}(t)} S_{\mathfrak{L}}.$$

Recall that the underlying graph of $(\mathbf{H}, W) \oplus (\mathbf{X}, S_{\mathfrak{L}})$ has at most $f(t') + |X| = O_{f(t)}(|X|)$ vertices and that its treewidth is bounded by $f(t') - t' + \mathbf{tw}(G[X]) \leq f(t) + t$. Therefore, according to Lemma 1, each $S_{\mathfrak{L}}$ can be computed in time $O_{f,t}(|X|)$. Moreover, as $|\mathsf{rep}_{\mathcal{A}}(t')| = O_{f,t}(1)$, we conclude that the set Y' can be computed in time $O_{f,t}(|X|)$.

Notice that $Y' \subseteq Y$ and that $|Y' \cap X| = |W| = \sum_{\mathfrak{L} \in \mathsf{rep}_{\mathcal{A}}(t')} |S_{\mathfrak{L}}| \leq |\mathsf{rep}_{\mathcal{A}}(t')| \cdot k = O_{f,t}(k)$. It also follows that $((G, Y), k)$ and $((G, Y'), k)$ are equivalent instances of $\Pi^{\alpha}_{\mathcal{A},\min}$ and this is so because the sets $S_{\mathfrak{L}}$ have been chosen so to represent every possible restriction in X of a solution of $\Pi^{\alpha}_{\mathcal{A},\min}$ on G (see [14, Lemma 5.3] for the complete argumentation).

Lemma 2 is already an important step towards a (bi)kernelization algorithm as it reduces the set of annotated vertices that can be inside the protrusions to a linear function of the parameter k. The next step is to "completely eliminate" the presence of annotated vertices in the "interior", i.e., $X \setminus \partial_G(X)$, of each flap X of a protrusion decomposition of the input graph.

Lemma 3. *Let $t \in \mathbb{N}$, G be a graph, $Y \subseteq V(G)$, and $\mathcal{P} = \{X_0, X_1, \ldots, X_\ell\}$ be a $(t \cdot k, t)$-protrusion-decomposition of G such that, for every $i \in [\ell]$, $|Y \cap X_i| \leq t \cdot k$. Then G has a $(O_t(k^2), O_t(1))$-protrusion-decomposition $\mathcal{P}'$ whose core contains Y as a subset. Moreover, $\mathcal{P}'$ can be computed in $O_t(|G|)$ steps.*

Proof (Proof (sketch).). The algorithm works, separately on each flap X of $\mathcal{P}$, on the tree decomposition $D = (T, \chi)$ of the graph induced by X and its neighborhood. In fact, we consider a decomposition $D = (T, \chi)$ of $G[N_G[X]]$ where T is a rooted binary tree and where each of its bags contains $N_G(X)$ and the root bag is $N_G(X)$. Notice that every bag of D has at most $2t + 1$ vertices. We declare nodes of D *dirty* as follows: for every $v \in Y$, mark as dirty the topmost bag containing v. This declares at most $t \cdot k$ of the nodes of D dirty. Next, proceed in a bottom up fashion (starting from the leaves of T), by declaring dirty each node that is a least common ancestor of two already dirty nodes. When this procedure finishes, we have less than $2t \cdot k$ dirty nodes. The new protrusion-decomposition is built by adding in the core X_0 of $\mathcal{P}$ the bags of all the nodes that have been declared dirty, for each flap of $\mathcal{P}$. This makes a new core X'_0 of $O_t(k^2)$ vertices. The connected components of $G \setminus X'_0$ can be organized to $O_t(k^2)$ subsets of the flaps of $\mathcal{P}$ such that, given the choice of the dirty vertices,

each of them has neighbors in at most two dirty bags. Therefore each of these subsets has at most $2(2t+1)$ neighbors and this permits to organize them to a $(O_t(k^2), O_t(1))$-protrusion-decomposition $\mathcal{P}'$, as required (see [14, Lemma 5.4] for more details).

Protrusion replacements. We next define the concept of replacing a protrusion X of G by another one. Let G be a graph G and let X be a t-protrusion of G where $|\partial_G(X)| = t' \leq t$. We set $G_X = G[X]$, $B = \partial_G(X)$ and we pick an arbitrary bijection $\rho : B \to [t']$. This gives a way to see the protrusion X as the t'-boundaried graph $\mathbf{G}_X = (G_X, B, \rho)$. We refer to $\mathbf{G}_X$ as *a boundaried version of* X. Notice that there are $t'!$ different boundaried versions of X, depending on the choice of ρ. Let now $\hat{\mathbf{G}} = (\hat{G}, \hat{B}, \hat{\rho})$ be a t'-boundaried graph that is compatible to $\mathbf{G}_X$. The graph occurring after *replacing* $\mathbf{G}_X$ *by* $\hat{\mathbf{G}}$ *in* G, denoted by $\mathsf{repl}(G, \mathbf{G}_X, \hat{\mathbf{G}})$, is the graph $\mathbf{F} \oplus \hat{\mathbf{G}}$, where $\mathbf{F} = (G \setminus (X \setminus B), B, \rho)$.

Restricted t-boundaried annotated graphs. We say that a t-boundaried annotated graph $(\mathbf{G}, Y)$ –where $\mathbf{G} = (G, B, \rho)$– is *restricted* if $Y \subseteq B$. Let $(\mathbf{G}_1, Y_1)$ and $(\mathbf{G}_2, Y_2)$ be two compatible restricted t-boundaried annotated graphs where $\mathbf{G}_i = (G_i, B_i, \rho_i), i \in [2]$. Observe that, because of our assumptions, the annotated vertices of $(\mathbf{G}_1, Y_1)$ and $(\mathbf{G}_2, Y_2)$ have boundary vertices corresponding to the same set of indices. Given a $t \in \mathbb{N}$, we say that $(\mathbf{G}_1, Y_1) \sim_{\mathcal{A},t} (\mathbf{G}_2, Y_2)$ if

$$\forall (S_1, S_2) \in 2^{Y_1} \times 2^{Y_2} \quad \Big(\rho_1(S_1) = \rho_2(S_2) \Rightarrow (\mathbf{G}_1, S_1) \equiv_{\mathcal{A},t} (\mathbf{G}_2, S_2)\Big).$$

Notice that $\sim_{\mathcal{A},t}$ is an equivalence relation on restricted t-boundaried annotated graphs. By picking a minimum-size t-boundaried annotated graph from each equivalence class, we set up a set of representatives that, from now on, we denote by $\overline{\mathsf{rep}}_{\mathcal{A}}(t)$. Notice that if $\mathcal{A} \in \mathsf{FI}(f)$ for some increasing function $f : \mathbb{N} \to \mathbb{N}$, then $|\overline{\mathsf{rep}}_{\mathcal{A}}(t)| = O_{f,t}(1)$ and the maximum size of a member of $\overline{\mathsf{rep}}_{\mathcal{A}}(t)$ is bounded by $O_{f,t}(1)$.

Lemma 4. *Let $f : \mathbb{N} \to \mathbb{N}$ be an increasing function, $\mathcal{A} \in \mathsf{FI}(f)$, $\mathsf{opt} \in \{\max, \min\}$, and $t \in \mathbb{N}$. There exists an algorithm that, given an instance $((G, Y), k)$ of $\Pi^{\alpha}_{\mathcal{A},\mathsf{opt}}$, a t-protrusion X where $X \cap Y \subseteq \partial_G(X)$, and a boundaried version $\mathbf{G}_X = (G_X, B, \rho)$ of X, outputs in time $O_{f,t}(|X|)$, a boundaried graph $\hat{\mathbf{G}}$ such that*

1. *$\hat{\mathbf{G}}$ is compatible with $\mathbf{G}_X$,*
2. *$|\hat{\mathbf{G}}| = O_{f,t}(1)$ and,*
3. *if $G' = \mathsf{repl}(G, \mathbf{G}_X, \hat{\mathbf{G}})$, then $((G', Y), k)$ is an equivalent instance of $\Pi^{\alpha}_{\mathcal{A},\mathsf{opt}}$.*

Proof (Proof (sketch).). Again we present only the case where $\mathsf{opt} = \min$ (see [14, Lemma 5.15] for the case where $\mathsf{opt} = \max$).

Let $Y_X = X \cap Y$ and $t_X = |\partial_G(X)| \leq t$. Recall that $Y_X \subseteq B$, therefore $\mathbf{G}_X = (G_X, B, \rho)$ is a restricted t_X-boundaried annotated graph. Clearly, there is some restricted t_X-boundaried annotated graph $(\hat{\mathbf{G}}, \hat{Y}_X) \in \overline{\mathsf{rep}}_{\mathcal{A}}(t_X)$ such that $(\mathbf{G}_X, Y_X) \sim_{\mathcal{A},t_X} (\hat{\mathbf{G}}, \hat{Y}_X)$, i.e., $(\hat{\mathbf{G}}, \hat{Y}_X)$ is a representative of $(\mathbf{G}_X, Y_X)$ with

respect to $\sim_{\mathcal{A},t_X}$. By the way $\hat{\mathbf{G}}$ is defined, it can be proven that $\hat{\mathbf{G}}$ is indeed a t_X-boundaried graph that can replace $\mathbf{G}_X$ towards creating an equivalent instance of $\Pi^{\alpha}_{\mathcal{A},\mathsf{opt}}$. For the full proof, see [14, Lemma 5.6].

It now remains to design an algorithm that, given a pair $(\mathbf{G}_X, Y_X)$, outputs its representative $(\hat{\mathbf{G}}, \hat{Y})$ in $O_{f,t}(|\mathbf{G}_X|)$ steps. For this we set up, for every $t' \in [0, 2t+1]$, the set $\mathcal{R}_{t'}$ containing every restricted t'-boundaried annotated graph $(\mathbf{G}, Y)$ where $|\mathbf{G}| \leq 2 \cdot f(2t+1)$. For every $t' \in [0, 2t+1]$, we next set up the function $\mathfrak{f}_{t'} : \mathcal{R}_{t'} \to \overline{\mathsf{rep}}_{\mathcal{A}}(t')$ mapping each $(\mathbf{G}, Y)$ in $\mathcal{R}_{t'}$ to its representative in $\overline{\mathsf{rep}}_{\mathcal{A}}(t')$, i.e., the member of $\overline{\mathsf{rep}}_{\mathcal{A}}(t')$ that is equivalent to $(\mathbf{G}, Y)$ with respect to the equivalence relation $\sim_{\mathcal{A},t'}$. Keep in mind that the function $\mathfrak{f}_{t'}$ does not depend on k. We clarify that $\mathfrak{f}_{t'}$ is hardcoded in the algorithm and is not computed on the fly.

The computation of $(\hat{\mathbf{G}}, \hat{Y})$ is done by standard dynamic programming on a tree decomposition $D = (T, \chi)$ of G_X whose tree is rooted on a node whose bag is B, where each node has at most two children, and where if a node has two children then the corresponding bags are the same as the bag of their parent. Every tree decomposition can be modified in linear time to one that satisfies these properties. This type of a decomposition is handy for performing dynamic programming and can be see as a simpler version of the concept of a *nice tree decomposition* (see [10, 78, 87] for more details on dynamic programming on nice tree decompositions). Each bag of D has at most $2t+1$ vertices as in the beginning of the proof of Lemma 3. For every node $i \in V(T)$, with $t_i = |\chi(i)|$, consider the restricted t_i-boundaried annotated graph $(\mathbf{G}_i, Y_i)$ where

1. the boundary of $\mathbf{G}_i$ is $\chi(i)$,
2. the underlying graph, say G_i, of $\mathbf{G}_i$ is the subgraph of G_X induced by $\chi(i)$ and all the vertices in bags of descendants of i in T, and
3. $Y_i = Y_X \cap V(G_i)$.

The dynamic programming algorithm computes, in a bottom-up fashion, for every bag $\chi(i)$ on t_i vertices, the representative $(\hat{\mathbf{G}}_i, \hat{Y}_i)$ of $(\mathbf{G}_i, Y_i)$ given that the representatives of the restricted boundaried graphs of the children of node i in T that have already been computed. This computation is done in $O_{f,t}(1)$ steps using the function $\mathfrak{f}_{t_i}$ and taking in mind that each i has at most two children, therefore $(\hat{\mathbf{G}}_i, \hat{Y}_i)$ is the application of $\mathfrak{f}_{t_i}$ to the "gluing" of the representatives corresponding to the children of i, that is a t_i-boundaried annotated graph whose boundaried graph has size at most $2 \cdot f(2t+1)$. In total $O(|G_X|)$ nodes are being processed, therefore the dynamic programming algorithm takes time $O_{f,t}(|G_X|)$, as required.

We are now in position to give the proof of Theorem 1.

Proof (Proof of Theorem 1). We present a bikernelization from $\Pi_{\mathcal{A},\mathsf{opt}}$ to $\Pi^{\alpha}_{\mathcal{A},\mathsf{opt}}$. Let $(G, k) \in \mathcal{G}_{\text{all}} \times \mathbb{N}$ be an input of $\Pi_{\mathcal{A},\mathsf{opt}}$. We fist use the time $\mathsf{T}^1(n)$ algorithm of the first condition in order to check whether $G \in \mathsf{dom}(\mathcal{A})$. If the answer is negative then $((G, Y), k)$ is a no-instance of $\Pi^{\alpha}_{\mathcal{A},\min}$ and we are done. In case of a positive answer, we know that $G \in \mathsf{dom}(\mathcal{A})$. This permits us to call the time

$\mathsf{T}^2(n)$ algorithm of the second condition that either outputs that $\mathbf{p}_{\mathcal{A},\mathsf{opt}}(G) > k_{\mathrm{opt}}$ or outputs a $(c \cdot k_{\mathsf{opt}}, c)$-protrusion-decomposition of G, where $k_{\mathrm{opt}} = k$ if $\mathsf{opt} = \min$ and $k_{\mathrm{opt}} = k - 1$ if $\mathsf{opt} = \max$. If $\mathbf{p}_{\mathcal{A},\mathsf{opt}}(G) > k_{\mathrm{opt}}$ we have that (G, k) is a no-instance of $\Pi_{\mathcal{A},\mathsf{opt}}$, in case $\mathsf{opt} = \min$, while it is a yes-instance of $\Pi_{\mathcal{A},\mathsf{opt}}$, in case $\mathsf{opt} = \max$. Depending on which case applies, the kernelization algorithm outputs a trivial no- or yes-instance of $\Pi^{\alpha}_{\mathcal{A},\mathsf{opt}}$.

Assume now that $\mathcal{P} = \{X_0, X_1, \ldots, X_\ell\}$ is a $(c \cdot k_{\mathsf{opt}}, c)$-protrusion-decomposition of G. We set $Y = V(G)$ and notice that (G, k) is a yes-instance of $\Pi_{\mathcal{A},\mathsf{opt}}$ iff $((G, Y), k)$ is a yes-instance of $\Pi^{\alpha}_{\mathcal{A},\min}$. By repetitively applying Lemma 2 for each flap of $\mathcal{P}$, we construct an equivalent instance $((G, Y'), k)$ of $\Pi^{\alpha}_{\mathcal{A},\min}$ where $|Y' \cap X_i| = O_{f,c}(k)$, for every $i \in [\ell]$. Then we apply Lemma 3 on $\mathcal{P}$ and construct a $(O_{f,c}(k^2), O_{f,c}(1))$-protrusion-decomposition $\mathcal{P}'$ whose core contains Y as a subset. Next, we repetitively apply Lemma 4 for each of the flaps of $\mathcal{P}'$ and construct an instance $((G', Y'), k)$ of $\Pi^{\alpha}_{\mathcal{A},\mathsf{opt}}$ that is equivalent to $((G, Y'), k)$ and therefore to $((G, Y), k)$ as well. That way, G' has a $(O_{f,c}(k^2), O_{f,c}(1))$-protrusion-decomposition where each flap contains at most $O_{f,c}(1)$ vertices. This implies that $|G'| = O_{f,c}(k^2)$. According to the running times of each of the three aforementioned lemmata, the construction of $((G', Y'), k)$ can be done in $O_{f,c}(|G|)$ steps.

Theorem 1 can be seen as a proper abstract version of Theorem 1.1 in [14]. A somehow stronger version of Theorem 1, that avoids the use of bikernels, can be derived in case $\mathsf{opt} = \min$. The proof is the same as the one of Theorem 1 (when $\mathsf{opt} = \min$) and is based on the additional observation that $\mathbf{p}_{\mathcal{A},\min}(G) > k_{\mathrm{opt}}$ implies that (G, k_{opt}) is a no-instance of $\Pi^{\alpha}_{\mathcal{A},\min}$. This gives rise to the following.

Theorem 3. *Let $f : \mathbb{N} \to \mathbb{N}$ be an increasing function, $\mathcal{A} \in \mathsf{FI}(f)$, and $c \in \mathbb{R}_{>0}$. If*

1. *$\mathsf{dom}(\mathcal{A})$ is computable in polynomial time $\mathsf{T}^1(n)$,*
2. *$(\mathcal{A}, \min)$ is c-protrusion-decomposable in polynomial time $\mathsf{T}^2(n)$,*

then $\Pi^{\alpha}_{\mathcal{A},\min}$ admits a polynomial size kernelization of size $k^{O_{f,c}(1)}$, running in polynomial time $\mathsf{T}^1(n) + \mathsf{T}^2(n) + O_{f,c}(n)$.

Admittedly, we do have a version of Theorem 3 when $\mathsf{opt} = \max$ as, in this case, we require that $\mathbf{p}_{\mathcal{A},\max}(G) > k_{\mathrm{opt}}$ implies that $(G, k_{\max})$ is a yes-instance of $\Pi^{\alpha}_{\mathcal{A},\max}$ and such an implication is not a consequence of the definition of the annotation version of a maximization problem.

We now arrive to the proof of Theorem 2.

Proof (Proof of Theorem 2). Let (G, k) be an instance of $\Pi_{\mathcal{A},\min}$ and let $((G', Y'), k)$ be an equivalent instance of $\Pi^{\alpha}_{\mathcal{A},\min}$, of size $k^{O_{f,c}(1)}$, constructed in time $\mathsf{T}^1(n) + \mathsf{T}^2(n) + O_{f,c}(n)$ by the application of the bikernelization algorithm of Theorem 1.

By the two last conditions we have that there is a polynomial time reduction from $\Pi^{\alpha}_{\mathcal{A},\mathsf{opt}}$ to $\Pi_{\mathcal{A},\mathsf{opt}}$. This means that there is a polynomial-time algorithm, i.e., an algorithm that runs in time $|G'|^{O(1)} = k^{O_{f,c}(1)}$, that can

transform $((G', Y'), k)$ to an instance $(\tilde{G}, \tilde{k})$ of $\Pi_{\mathcal{A},\mathsf{opt}}$ such that $((G', Y'), k)$ is a yes-instance of $\Pi^{\alpha}_{\mathcal{A},\min}$ iff $(\tilde{G}, \tilde{k})$ is a yes-instance of $\Pi_{\mathcal{A},\min}$. Notice that $\max\{|\hat{G}|, \hat{k}\} = k^{O_{f,c}(1)}$ and that (G, k) and $(\tilde{G}, \tilde{k})$ are equivalent instances of $\Pi_{\mathcal{A},\mathsf{opt}}$, as required.

3.2 Meta-kernels for subset properties with finite integer index

Our second meta-algorithmic result is based on the notion of finite integer index.

Finite integer index. Two t-boundaried graphs $\mathbf{G}_1$ and $\mathbf{G}_2$ are *compatible* if the t-boundaried annotated graphs $(\mathbf{G}_1, \emptyset)$ and $(\mathbf{G}_2, \emptyset)$ are compatible. Given two t-boundaried graphs $\mathbf{G}_1$ and $\mathbf{G}_2$ we define $\mathbf{G}_1 \oplus \mathbf{G}_2$ as the graph of the pair $(\mathbf{G}_1, \emptyset) \oplus (\mathbf{G}_2, \emptyset)$.

Let $(\mathcal{A}, \mathsf{opt})$ be a defining pair and $t \in \mathbb{N}$. Given two compatible t-boundaried graphs $\mathbf{G}_1, \mathbf{G}_2$, we say that $\mathbf{G}_1 \approx_{\mathcal{A},\mathsf{opt},t} \mathbf{G}_2$ if there exists a constant $c_{\mathbf{G}_1,\mathbf{G}_2} \in \mathbb{Z}$, depending on $\mathbf{G}_1$ and $\mathbf{G}_2$, such that for every t-boundaried graph $\mathbf{F}$ that is compatible with $\mathbf{G}_1$, it holds that

$$\mathbf{p}_{\mathcal{A},\mathsf{opt}}(\mathbf{F} \oplus \mathbf{G}_2) = \mathbf{p}_{\mathcal{A},\mathsf{opt}}(\mathbf{F} \oplus \mathbf{G}_1) + c_{\mathbf{G}_1,\mathbf{G}_2}.$$

For completeness, if in the above definition, for some $i \in [2]$, the graph $\mathbf{F} \oplus \mathbf{G}_i \notin \mathsf{dom}(\mathbf{p}_{\mathcal{A},\mathsf{opt}})$, we assume that $\mathbf{p}(\mathbf{G}_i \oplus \mathbf{F}) = \infty$. Observe that c might be negative in the above definition. In fact, $c_{\mathbf{G}_1,\mathbf{G}_2} = -c_{\mathbf{G}_2,\mathbf{G}_1}$. Note that the relation $\approx_{\mathcal{A},\mathsf{opt},t}$ is an equivalence relation. We set up, for every $t \in \mathbb{N}$, a set $\widetilde{\mathsf{rep}}_{\mathcal{A},\mathsf{opt}}(t)$ of representatives of the relation $\approx_{\mathcal{A},\mathsf{opt},t}$ by picking one minimum-size representative for each of its equivalence classes.

Given an increasing function $f : \mathbb{N} \to \mathbb{N}$, we define $\mathsf{FII}(f)$ as the set of all defining pairs $(\mathcal{A}, \mathsf{opt})$ where for every $t \in \mathbb{N}$, the size of every t-boundaried graph in $\widetilde{\mathsf{rep}}_{\mathcal{A},\mathsf{opt}}(t)$ is upper bounded by $f(t)$. We say that the defining pair $(\mathcal{A}, \mathsf{opt})$ has *finite integer index* if it belongs in $\mathsf{FII}(f)$, for some increasing function $f : \mathbb{N} \to \mathbb{N}$. The defining pair $(\mathcal{A}, \mathsf{opt})$ has finite integer index if and only if $\approx_{\mathcal{A},\mathsf{opt},t}$ has a finite number of equivalence classes, for every $t \in \mathbb{N}$.

Notice that the notion of finite integer index concerns different objects than the one of finite index. Having finite index is a property of vertex-subset properties, while having finite integer index is a property of defining pairs.

The notion of finite integer index was introduced in the thesis of Babette van Antwerpen-de Fluiter [44], see also [5, 12, 13, 43]. Our second meta-algorithmic result, corresponding to theorem 1.3 of [14], is the following.

Theorem 4. *Let $f : \mathbb{N} \to \mathbb{N}$ be an increasing function, $(\mathcal{A}, \mathsf{opt}) \in \mathsf{FII}(f)$, and $c \in \mathbb{R}_{>0}$. If*

1. *$\mathsf{dom}(\mathcal{A})$ is computable in polynomial time $\mathsf{T}^1(n)$ and*
2. *$(\mathcal{A}, \mathsf{opt})$ is c-protrusion-decomposable in polynomial time $\mathsf{T}^2(n)$.*

then $\Pi_{\mathcal{A},\mathsf{opt}}$ admits a linear kernelization of size $O_{f,c}(k)$, running in polynomial time $\mathsf{T}^1(n) + \mathsf{T}^2(n) + O_{f,c}(n)$.

Proof (Proof (sketch).). The idea of the proof follows from the fact that if X is a protrusion of G, the t-boundaried graph $\mathbf{G}_X$ is a boundaried version of X, $\hat{\mathbf{G}}$ is the representative of $\mathbf{G}_X$ in $\widetilde{\mathsf{rep}}_{\mathcal{A}}(\mathsf{opt}, t)$, $G' = \mathsf{repl}(G, \mathbf{G}_X, \hat{\mathbf{G}})$, and $k' = k + c_{\mathbf{G}_i, \hat{\mathbf{G}}_i}$, then (G, k) and (G', k') are equivalent instances of $\Pi_{\mathcal{A},\mathsf{opt}}$. The proof of this fact follows directly from the definition of the equivalence relation $\approx_{\mathcal{A},\mathsf{opt},t}$ (see [14, Lemma 5.18] for the details).

As in the begining of the proof of Theorem 2 we can assume that we have a protrusion-decomposition $\mathcal{P} = \{X_0, X_1, \ldots, X_\ell\}$ of G, that has been constructed in time $\mathsf{T}^1(n) + \mathsf{T}^2(n)$ (otherwise a direct answer can be derived and the algorithm outputs a trivial equivalent instance). The algorithm performs dynamic programming on the special type of tree decomposition that we used in the proof of Lemma 4. For every flap X of $\mathcal{P}$, the dynamic programming runs on a boundaried version $\mathbf{G}_X = (G_X, B, \rho)$ of X and detects a node i where the t_i-boundaried graph $\mathbf{G}_i$ (defined as in the proof of Theorem 2) has size more than $f(t)$ but also has size $O(f(t))$. The upper bound permits the detection of i in $O_{f,c}(1)$ steps. The lower bound permits the replacement in G of $\mathbf{G}_i$ by its representative $\hat{\mathbf{G}}_i$ in $\widetilde{\mathsf{rep}}_{\mathcal{A},\mathsf{opt}}(t)$ that has smaller size. Consider the pair (G', k'), where G' is the result of the protrusion replacement and $k' = k + c_{\mathbf{G}_i, \hat{\mathbf{G}}_i}$ and recall that (G, k) and (G', k') are equivalent instances of $\Pi_{\mathcal{A},\mathsf{opt}}$ where $|G'| < |G|$. By performing such replacements bottom-up and updating the protrusion-decomposition accordingly, we keep creating equivalent instances until all the flaps in the protrusion-decomposition of the resulting graph are of size upper-bounded by $2^{f(t)}$. This ends up, in time $O_{f,c}(n)$, with an equivalent instance (G', k') whose size is a linear function of k.

Notice that the first condition of both Theorem 2 and Theorem 4 require the polynomial computability of the domain of $\mathcal{A}$. In some cases, variants of these theorems are ignoring (or ommiting) this condition as it is either obvious or the problem is stated so that containment to $\mathcal{A}$ is a promise condition.

4 Consequences

In this section we deal with the applications of Theorem 2 and Theorem 4. Notice that both these theorems contain two types of requirements. The first is that the vertex-subset property (resp. defining pair) has finite integer index (resp. finite integer index), the second is that the problem is protrusion decomposable. The first condition will be linked to the descriptive complexity of the problem while the second will be linked to certain combinatorial properties of its inputs, i.e., the domain of $\mathcal{A}$.

4.1 Counting Monadic Second Order Logic

There is a wide variety of vertex-subset problems generated by defining pairs. Typically, they can be defined using logical sentences.

CMSOL. The syntax of Counting Monadic Second Order Logic (CMSOL) on graphs includes the logical connectives $\vee$, $\wedge$, $\neg$, $\rightarrow$, $\leftrightarrow$, variables for vertices, edges, sets of vertices, and sets of edges, the quantifiers $\forall$, $\exists$ that can be applied to these variables, and the following six predicates:

1. $\mathbf{vin} : V(G) \times 2^{V(G)} \rightarrow \{\mathtt{T}, \mathtt{F}\}$, where $\mathbf{vin}(v, S) = \mathtt{T}$ iff v is a vertex of S,
2. $\mathbf{ein} : E(G) \times 2^{E(G)} \rightarrow \{\mathtt{T}, \mathtt{F}\}$, where $\mathbf{ein}(e, F) = \mathtt{T}$ iff e is an edge of F,
3. $\mathbf{inc} : V(G) \times E(G) \rightarrow \{\mathtt{T}, \mathtt{F}\}$, where $\mathbf{inc}(v, e) = \mathtt{T}$ iff v is an endpoint of e,
4. $\mathbf{adj} : (V(G))^2 \rightarrow \{\mathtt{T}, \mathtt{F}\}$, where $\mathbf{adj}(v, u) = \mathtt{T}$ iff v and u are distinct endpoints of an edge,
5. $\mathbf{eq} : (V(G))^2 \rightarrow \{\mathtt{T}, \mathtt{F}\}$, where $\mathbf{eq}(v, u) = \mathtt{T}$ iff v and u are equal,
6. $\mathbf{card}_{q,r} : V(G) \rightarrow \{\mathtt{T}, \mathtt{F}\}$, where $\mathbf{card}_{q,r}(S) = \mathtt{T}$ iff $|S| \equiv q \pmod r$, where r, q are fixed integers such that $0 \leq q < r$ and $r \geq 2$.

We use variants of the symbols v, e, S, and F in order to denote variants of vertices, edges, vertex sets, and edge sets respectively. We refer to [6, 21, 23] for a detailed introduction on CMSO. Given a sentence ϕ, we denote its length by $|\phi|$.

Some examples. We may consider CMSOL sentences that are evaluated either on graphs or on annotated graphs. For instance, if

$$\phi = \forall v_1 \ \big(\mathbf{vin}(v_1, S) \ \vee \ \exists v_2 \ (\mathbf{adj}(v_1, v_2) \wedge \mathbf{vin}(v_2, S))\big),$$

then $(G, S) \models \phi$ iff S is a dominating set of G. Moreover, if

$$\phi = \forall S_1, S_2 \ \Big(\Big(\exists x \ \mathbf{vin}(x, S_1) \wedge \exists x \ \mathbf{vin}(x, S_2) \wedge$$

$$\forall v \ \big(\big(\mathbf{vin}(v, S_1) \wedge \neg\mathbf{vin}(v, S_2)\big) \vee \big(\mathbf{vin}(v, S_2) \wedge \neg\mathbf{vin}(v, S_1)\big)\big)\Big) \rightarrow$$

$$\Big(\exists v_1, v_2, e \ \big(\mathbf{vin}(v_1, S_1) \wedge \mathbf{vin}(v_2, S_2) \wedge \mathbf{inc}(v_1, e) \wedge \mathbf{inc}(v_2, e)\big)\Big)\Big),$$

then $G \models \phi$ iff G is connected.

Problems defined by sentences. Given a CMSOL sentence ψ on graphs, we define the graph class

$$\mathcal{G}_\psi = \{G \mid G \models \phi\}.$$

Moreover, given a CMSOL sentence ϕ on annotated graphs, we define the vertex-subset property

$$\mathcal{A}_\phi = \{(G, S) \mid (G, S) \models \phi\}.$$

Given a CMSOL sentence ϕ on annotated graphs and an $\mathsf{opt} \in \{\min, \max\}$, we use $\Pi_{\phi,\mathsf{opt}}$ and $\Pi^{\alpha}_{\phi,\mathsf{opt}}$ as shortcuts for the vertex-subset problem $\Pi_{\mathcal{A}_\phi,\mathsf{opt}}$ and its annotated version $\Pi^{\alpha}_{\mathcal{A}_\phi,\mathsf{opt}}$ respectively. Also, given a CMSOL sentence ψ on graphs and a CMSOL sentence ϕ on annotated graphs, we define $\phi \Cap \psi$ as the CMSOL sentence where

$$(G, S) \models \phi \Cap \psi \text{ iff } (G, S) \models \phi \text{ and } G \models \psi.$$

By admitting that shortcuts may create inflation of notation, we notice that $\mathcal{A}_\phi \Cap \mathcal{G}_\psi = \mathcal{A}_{\phi \Cap \psi}$ and that $\Pi_{\phi,\mathsf{opt}} \Cap \mathcal{G}_\psi = \Pi_{\mathcal{A}_\phi \Cap \mathcal{G}_\psi,\mathsf{opt}} = \Pi_{\mathcal{A}_{\phi \Cap \psi},\mathsf{opt}} = \Pi_{\phi \Cap \psi,\mathsf{opt}}$.

4.2 Properties of defining pairs

We now come to the combinatorial properties of defining pairs. In our setting, these properties condition the domain of the vertex subset property.

Treewidth-modulability. Given a defining pair $(\mathcal{A}, \mathsf{opt})$ and a $c \in \mathbb{R}_{>0}$, we say that the defining pair $(\mathcal{A}, \mathsf{opt})$ is *c-treewidth-modulable* if, for every $(G, k) \in \mathsf{dom}(\mathcal{A}) \times \mathbb{N}$, it holds that

$$\mathbf{p}_{\mathcal{A},\mathsf{opt}}(G) \leq k \Rightarrow \exists S \subseteq V(G) : |S| \leq c \cdot k \ \wedge \ \mathbf{tw}(G \setminus S) \leq c.$$

Minors, contractions, and topological minors. Given a graph G, we say that a graph H is a *contraction* of G if a graph isomorphic to H can be obtained from G after contracting edges. We also say that H is *a minor* of G if it is the contraction of some subgraph of G. Finally, we say that H is a *topological minor* of G if some subdivision of H is isomorphic to a subgraph of G (a *subdivision* of H is any graph obtained by replacing edges by paths with the same endpoints).

Given a finite set of graphs $\mathcal{H}$, we denote by $\mathcal{C}_{\mathcal{H}}$ the class of graphs that do not contain any of the graphs in $\mathcal{H}$ as a contraction, by $\mathcal{T}_{\mathcal{H}}$ the class of graphs that do not contain any of the graphs in $\mathcal{H}$ as a topological minor, and by $\mathcal{M}_{\mathcal{H}}$ the class of graphs that do not contain any of the graphs in $\mathcal{H}$ as a minor. Notice that, for every finite set of graphs $\mathcal{H}$, $\mathcal{M}_{\mathcal{H}} \subseteq \mathcal{T}_{\mathcal{H}}$ and $\mathcal{M}_{\mathcal{H}} \subseteq \mathcal{C}_{\mathcal{H}}$.

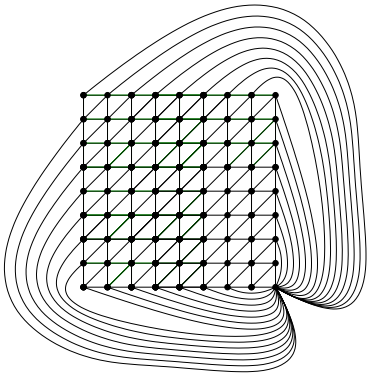

Fig. 1. Graph Γ_9.

SQGM and SQGC properties of graph classes. Let $\mathcal{G} \subseteq \mathcal{G}_{\text{all}}$, $\lambda \in \mathbb{R}_{>0}$, and let c be a real number in the interval $[1, 2)$. We say that $\mathcal{G}$ has the *sub-quadratic grid minor property* (SQGM property, for short) *for* λ *and* c, if

$$\forall k \in \mathbb{N} \ \forall G \in \mathcal{M}_{\{\boxplus_k\}} \ \ \mathbf{tw}(G) \leq \lambda \cdot k^c.$$

Also, we say that a graph class $\mathcal{G}$ has the *sub-quadratic graph contraction property* (SQGC property for short) *for* λ *and* c, if

$$\forall k \in \mathbb{N}\ \forall G \in \mathcal{C}_{\{\Gamma_k\}}\ \ \mathbf{tw}(G) \leq \lambda \cdot k^c.$$

We denote by $\texttt{SQGM}(\lambda, c)$ (resp. $\texttt{SQGC}(\lambda, c)$) the set of all graph classes that have the SQGM (resp, SQGC) property for λ and c.

The most simple example of a graph class that has the above properties is the class of planar graphs, that belongs in $\texttt{SQGC}(4.2, 1)$ [68]. For more general classes of graphs satisfying the SQGM and the SQGC properties, see [7,31,48,52,53,64].

Bidimensonality. Given a $k \in \mathbb{N}_{\geq 1}$, a $(k \times k)$-*grid* is the graph $\boxplus_k$ where $V(\boxplus_k) = [k]^2$ and $E(\boxplus_k) = \{\{(x, y), (x', y')\} \mid |x - x'| + |y - y'| = 1\}$. The *perimeter* of $\boxplus_k$, denoted by $P(\boxplus_k)$, is the set containing all the vertices of $\boxplus_k$ that have degree smaller than 4. The *uniformly triangulated grid* $(k \times k)$*-grid* is the graph Γ_k where $V(\Gamma_k) = [k]^2$ and

$$\begin{aligned} E(\Gamma_k) = E(\boxplus_k) \cup \\ \{(x+1, y), (x, y+1) \mid (x, y) \in [t-1]^2\} \cup \\ \{\{(k, k), (a, b)\} \mid (a, b) \in P(\boxplus_k) \setminus \{(k, k)\}\}). \end{aligned}$$

For a drawing of Γ_9, see Figure 1.

Given two real functions f and g, we use the term $f \gtrsim g$ to denote that $f(x) \geq g(x) - o(g(x))$.

Let $(\mathcal{A}, \mathsf{opt})$ be a defining pair. We say that $(\mathcal{A}, \mathsf{opt})$ is *minor closed* (resp. *contraction closed*) if for every two graphs $G_1, G_2 \in \mathsf{dom}(\mathcal{A})$, it holds that if G_1 is a minor (resp. contraction) of G_2, then $\mathbf{p}_{\mathcal{A},\mathsf{opt}}(G_1) \leq \mathbf{p}_{\mathcal{A},\mathsf{opt}}(G_2)$.

Given a $c \in \mathbb{R}_{>0}$, we say that $(\mathcal{A}, \mathsf{opt})$ is *c-minor-bidimensional* (resp. *c-contraction-bidimensional*) if it is minor-closed (resp. contraction-closed) and $\mathbf{p}_{\mathcal{A},\mathsf{opt}}(\boxplus_k) \gtrsim ck^2$ (resp. $\mathbf{p}_{\mathcal{A},\mathsf{opt}}(\Gamma_k) \gtrsim ck^2$).

Bidimensonality was introduced in [30] and has been the combinatorial base of several algorthmic results concerning subexponential parameterized algorithms [29, 30,34,47], the design of efficient polynomial-time approximation schemes [32,51, 53], and, of course, kernelization [54,55,76], see also [28,33,46,73,85,86].

Separability. Let $(\mathcal{A}, \mathsf{opt})$ be a defining pair and let $f : \mathbb{N} \to \mathbb{N}$. We say that $(\mathcal{A}, \mathsf{opt})$ is *f-separable* if for every graph $G \in \mathsf{dom}(\mathcal{A})$, every $S \in \mathsf{sol}_{\mathcal{A},\mathsf{opt}}(G)$, and every $L \subseteq V(G)$, it holds that

$$|S \cap L| - f(t) \leq \mathbf{p}_{\mathcal{A},\mathsf{opt}}(G[L]) \leq |S \cap L| + f(t),$$

where $t = |\partial_G(L)|$. Given some $c \in \mathbb{R}_{>0}$, we say that the defining pair $(\mathcal{A}, \mathsf{opt})$ is *c-linearly-separable* if it is *f-separable* for the function $f : \mathbb{N} \to \mathbb{N}$, defined so that $f(x) = c \cdot x$.

The notion of separability has been introduced in [54, 55], while a similar notion had earlier appeared in [32].

4.3 Theorems on properties of defining pairs

Conditions for proving finite index and finite integer index. The following celebrated result, widely known as Courcelle's theorem, is the standard way to prove that a vertex-subset property has finite index.

Proposition 1. *For every CMSOL sentence ϕ, there is an increasing function $f : \mathbb{N} \to \mathbb{N}$ such that $\mathcal{A}_\phi \in \mathsf{FI}(f)$.*

Proposition 1 has appeared in many forms and with different proofs, see [2,6,17, 21,23,35]. For a proof of a more general version that the one in Proposition 1, see [14, Lemma 3.2].

The next result gives a criterion for proving that a defining pair has finite integer index.

Proposition 2. *For every CMSOL sentence ϕ, $\mathsf{opt} \in \{\min, \max\}$, and every function $f : \mathbb{N} \to \mathbb{N}$, if the defining pair $(\mathcal{A}_\phi, \mathsf{opt})$ is f-separable, then there is some increasing function f' such that $(\mathcal{A}, \mathsf{opt}) \in \mathsf{FII}(f')$.*

Proposition 2 appeard in [55]. It provides an easy way to prove that a defining pair has the FII property. For proving FII, an alternative to separability, called strong monotonicity, had already appeared in [14, Section 7].

Conditions for proving protrusion-decomposability. In both Theorems 1 and 3, the second condition is protrusion decomposability. This condition can be implied by other more easy to verify combinatorial conditions. The first one is treewidth-modulability combined with the exclusion of some graph as topological minor.

Theorem 5. *Let ϕ be a CMSOL sentence on annotated graphs, $h \in \mathbb{N}_{\geq 1}$, $c \in \mathbb{R}_{>0}$, and $\mathcal{H}$ be a set of graphs, each of size at most h. If $(\mathcal{A}_\phi, \mathsf{opt})$ is a c-treewidth-modulable defining pair, then $(\mathcal{A}_\phi \Cap \mathcal{T}_\mathcal{H}, \mathsf{opt})$ is $O_{|\phi|,h,c}(1)$-protrusion-decomposable in time $O_{|\phi|,h,c}(n^2)$.*

The proof of 5 is implicit in [76] and is presented in more detail in [77].

We should stress that the quadratic-time protrusion decomposability of the above result can be improved to a linear one by combining the results of [55, 76,77]. We avoid presenting the proof of this here, as it is lengthy and requires the introduction of several concepts from [49] and [50] such as *solution lifting*, *protrusion cover*, and *explicit representation of subgraphs* (the bulk of the arguments has already been exposed in [57, Subsection 15.6]).

The main combinatorial condition in Theorem 5 is protrusion-decomposability. The following result appeared in [55] and reveals how this condition can be implied from other, more easy to check, conditions.

Theorem 6. *Let $\lambda, c, c' \in \mathbb{R}_{>0}$, and let c'' be a real number in the interval $[1, 2)$, and let $f : \mathbb{N} \to \mathbb{N}$ be an increasing function. If $\mathcal{G} \in \mathtt{SQGM}(c'', \lambda)$ (resp. $\mathcal{G} \in \mathtt{SQGC}(c'', \lambda)$) and $(\mathcal{A}, \mathsf{opt})$ is a defining pair that is c-minor-bidimensional (resp. c-contraction-bidimensional) and c'-linearly separable, then $(\mathcal{A} \Cap \mathcal{G}, \mathsf{opt})$ is $O_{\lambda,c,c',c''}(1)$-treewidth-modulable.*

4.4 Applications

In this section we present the consequences of Theorem 2 and Theorem 4 under the light of the logical and combinatorial conditions of subsection 4.3. We expose them as two corollaries, corresponding to Theorem 2 and Theorem 4 respectively.

Corollary 1. *Let ϕ be a CMSOL sentence on annotated graphs, $h \in \mathbb{N}_{\geq 1}$, $c \in \mathbb{R}_{>0}$, $\mathcal{H}$ be a set of graphs, each of size at most h, and $\mathsf{opt} \in \{\min, \max\}$. If*

1. *$\mathsf{dom}(\mathcal{A}_\phi)$ is computable in polynomial time $\mathsf{T}(n)$,*
2. *$(\mathcal{A}_\phi, \mathsf{opt})$ is c-treewidth-modulable,*
3. *$\Pi_{\phi,\mathsf{opt}} \Cap \mathcal{T}_{\mathcal{H}}$ is NP-hard, and $\Pi^{\alpha}_{\phi,\mathsf{opt}} \in \mathsf{NP}$,*

then $\Pi_{\phi,\mathsf{opt}} \Cap \mathcal{T}_{\mathcal{H}}$ admits a polynomial kernelization of size $k^{O_{|\phi|,h,c}(1)}$, running in time $\mathsf{T}^1(n) + O_h(n^3) + O_{|\phi|,h,c}(n^2) + k^{O_{|\phi|,h,c}(1)}$.

If Condition 3 is replaced by the following:

3′. $(\mathcal{A}_\phi, \mathsf{opt})$ is f-separable, for some function $f : \mathbb{N} \to \mathbb{N}$,

then $\Pi_{\phi,\mathsf{opt}} \Cap \mathcal{T}_{\mathcal{H}}$ admits a linear size kernelization of size $O_{f,|\phi|,h,c}(k)$, running in time $\mathsf{T}^1(n) + O_h(n^3) + O_{f,|\phi|,h,c}(n^2)$.

Proof. From Theorem 5, $(\mathcal{A}_\phi \Cap \mathcal{T}_{\mathcal{H}}, \mathsf{opt})$ is c'''-protrusion-decomposable in time $\mathsf{T}^2(n) = O_{|\phi|,h,c}(n^2)$ for some $c' = O_{|\phi|,h,c}(1)$.

As topological minor containment can be expressed in monadic second order logic, there is some $\psi_{\mathcal{H}}$ such that $\mathcal{T}_{\mathcal{H}} = \mathcal{G}_{\psi_{\mathcal{H}}}$. Recall that $\phi \Cap \psi_{\mathcal{H}}$ is a CMSOL sentence on annotated graphs. Also $\mathcal{A}_\phi \Cap \mathcal{G}_{\psi_{\mathcal{H}}} = \mathcal{A}_{\phi \Cap \psi_{\mathcal{H}}}$ and $\Pi_{\phi,\mathsf{opt}} \Cap \mathcal{G}_\psi = \Pi_{\mathcal{A}_\phi \Cap \mathcal{G}_\psi,\mathsf{opt}}$, therefore $\Pi_{\phi,\mathsf{opt}} \Cap \mathcal{T}_{\mathcal{H}} = \Pi_{\phi \Cap \psi_{\mathcal{H}},\mathsf{opt}}$.

It was proved in [66], that $\mathcal{T}_{\mathcal{H}}$ is computable in time $O_h(n^3)$. Therefore $\mathcal{A}_{\phi \Cap \psi_{\mathcal{H}}}$ is computable in time $\mathsf{T}^1(n) = \mathsf{T}(n) + O_h(n^3)$. Notice also that if $\Pi^{\alpha}_{\phi,\mathsf{opt}} \in \mathsf{NP}$, then also $\Pi^{\alpha}_{\phi,\mathsf{opt}} \Cap \mathcal{T}_{\mathcal{H}} \in \mathsf{NP}$. The result follows by applying Theorem 2 and Theorem 4 for the defining pair $(\phi \Cap \psi_{\mathcal{H}}, \mathsf{opt})$ and the constant c'.

If now in the second version of Corollary 1, we deduce treewidth-modulability by using the conditions of Theorem 6, we derive the following.

Corollary 2. *Let ϕ be a CMSOL sentence on annotated graphs, $h \in \mathbb{N}_{\geq 1}$, $\lambda, c, c' \in \mathbb{R}_{>0}$, $c'' \in [1, 2)$, $\mathcal{H}$ be a set of graphs, each of size at most h, and $\mathsf{opt} \in \{\min, \max\}$. If*

1. *$\mathsf{dom}(\mathcal{A}_\phi)$ is computable in time $\mathsf{T}(n)$,*
2. *$(\mathcal{A}_\phi, \mathsf{opt})$ is c'-linearly separable,*
3. *$(\mathcal{A}_\phi, \mathsf{opt})$ is c-minor bidimensional (resp. c-minor bidimensional), and*
4. *$\mathcal{T}_{\mathcal{H}} \in \mathtt{SQGM}(c'', \lambda)$ (resp. $\mathcal{T}_{\mathcal{H}} \in \mathtt{SQGM}(c'', \lambda)$),*

then $\Pi_{\phi,\mathsf{opt}} \Cap \mathcal{T}_{\mathcal{H}}$ admits a linear size kernelization running in time $\mathsf{T}(n) + O_h(n^3) + O_{|\phi|,h,\lambda,c,c',c''}(n^2)$.

Concerning the applicability of Corollary 2, we stress that for every finite set of graphs $\mathcal{Z}$, each of size at most z, there is an $\mathcal{H}$ such that $\mathcal{T}_{\mathcal{H}} = \mathcal{M}_{\mathcal{Z}}$. Moreover, there is a constant λ_z such that $\mathcal{M}_{\mathcal{Z}} \in \texttt{SQGM}(\lambda_z, 1)$ [31]. This implies that the minor version of the fourth condition of Corollary 2 holds if we replace $\mathcal{T}_{\mathcal{H}}$ by any non-trivial minor-closed graph class (taking into account the Robertson & Seymour Theorem [84]). For the contraction version, assume additionally that $\mathcal{Z}$ contains at least one apex graph (an *apex* graph is a graph containing a vertex whose removal creates a planar graph). For every such $\mathcal{Z}$, it holds that $\mathcal{M}_{\mathcal{Z}} \in \texttt{SQGC}(\lambda_h, 1)$, for some constant λ_h [48]. This implies that the contraction version of the fourth condition of Corollary 2 holds if we replace $\mathcal{T}_{\mathcal{H}}$ by $\mathcal{M}_{\mathcal{Z}}$ and pick $\mathcal{Z}$ to be any finite set containing at least one apex graph.

Acknowledgements: I wish to whole-heartedly thank *Professor Hans L. Bodlaender* for being the one who *«told me a little but he taught me a lot»*.

References

1. Abrahamson, K.A., Downey, R.G., Fellows, M.R.: Fixed-parameter tractability and completeness. IV. On completeness for W[P] and PSPACE analogues. Annals of Pure and Applied Logic **73**(3), 235–276 (1995)
2. Abrahamson, K.R., Fellows, M.R.: Finite automata, bounded treewidth and well-quasiordering. In: Robertson, N., Seymour, P.D. (eds.) AMS Summer Workshop on Graph Minors, Graph Structure Theory, Contemporary Mathematics vol. 147, pp. 539–564. American Mathematical Society (1993)
3. Alber, J., Betzler, N., Niedermeier, R.: Experiments on data reduction for optimal domination in networks. Annals OR **146**(1), 105–117 (2006)
4. Alber, J., Fellows, M.R., Niedermeier, R.: Polynomial-time data reduction for dominating set. J. Assoc. Comput. Mach. **51**(3), 363–384 (2004)
5. Arnborg, S., Courcelle, B., Proskurowski, A., Seese, D.: An algebraic theory of graph reduction. J. ACM **40**, 1134–1164 (1993)
6. Arnborg, S., Lagergren, J., Seese, D.: Easy problems for tree-decomposable graphs. Journal of Algorithms **12**, 308–340 (1991)
7. Baste, J., Thilikos, D.M.: Contraction-bidimensionality of geometric intersection graphs. In: 12th International Symposium on Parameterized and Exact Computation, IPEC 2017, September 6-8, 2017, Vienna, Austria. pp. 5:1–5:13 (2017)
8. Bodlaender, H.L., Fomin, F.V., Lokshtanov, D., Penninkx, E., Saurabh, S., Thilikos, D.M.: (meta) Kernelization. In: FOCS 2009. pp. 629–638. IEEE (2009)
9. Bodlaender, H.L.: On reduction algorithms for graphs with small treewidth. In: van Leeuwen, J. (ed.) Graph-Theoretic Concepts in Computer Science, 19th International Workshop, WG '93, Utrecht, The Netherlands, June 16-18, 1993, Proceedings. Lecture Notes in Computer Science, vol. 790, pp. 45–56. Springer (1993). https://doi.org/10.1007/3-540-57899-4_40, https://doi.org/10.1007/3-540-57899-4_40
10. Bodlaender, H.L.: A partial k-arboretum of graphs with bounded treewidth. Theoret. Comput. Sci. **209**(1-2), 1–45 (1998)
11. Bodlaender, H.L., Downey, R.G., Fellows, M.R., Hermelin, D.: On problems without polynomial kernels. J. Comput. Syst. Sci. **75**, 423–434 (December 2009). https://doi.org/10.1016/j.jcss.2009.04.001, http://portal.acm.org/citation.cfm?id=1628322.1628467

12. Bodlaender, H.L., van Antwerpen-de Fluiter, B.: Reduction algorithms for constructing solutions in graphs with small treewidth. In: Cai, J., Wong, C.K. (eds.) Computing and Combinatorics, Second Annual International Conference, COCOON '96, Hong Kong, June 17-19, 1996, Proceedings. Lecture Notes in Computer Science, vol. 1090, pp. 199–208. Springer (1996). https://doi.org/10.1007/3-540-61332-3_153, https://doi.org/10.1007/3-540-61332-3_153
13. Bodlaender, H.L., van Antwerpen-de Fluiter, B.: Reduction algorithms for graphs of small treewidth. Information and Computation **167**, 86–119 (2001)
14. Bodlaender, H.L., Fomin, F.V., Lokshtanov, D., Penninkx, E., Saurabh, S., Thilikos, D.M.: (Meta) Kernelization. J. ACM **63**(5), 44:1–44:69 (2016)
15. Bodlaender, H.L., Penninkx, E.: A linear kernel for planar feedback vertex set. In: Proccedings of the 3rd International Workshop on Exact and Parameterized Computation (IWPEC 2008), LNCS, vol. 5018, pp. 160–171. Springer, Berlin (2008)
16. Bodlaender, H.L., Penninkx, E., Tan, R.B.: A linear kernel for the k-disjoint cycle problem on planar graphs. In: 19th International Symposium on Algorithms and Computation (ISAAC), LNCS, vol. 5369, pp. 306–317. Springer (2008)
17. Borie, R.B., Parker, R.G., Tovey, C.A.: Automatic generation of linear-time algorithms from predicate calculus descriptions of problems on recursively constructed graph families. Algorithmica **7**, 555–581 (1992)
18. Chen, J., Fernau, H., Kanj, I.A., Xia, G.: Parametric duality and kernelization: Lower bounds and upper bounds on kernel size. SIAM J. Comput. **37**(4), 1077–1106 (2007)
19. Chen, J., Kanj, I.A., Jia, W.: Vertex cover: further observations and further improvements. J. Algorithms **41**(2), 280–301 (2001)
20. Courcelle, B.: The monadic second-order logic of graphs. III. Tree-decompositions, minors and complexity issues. RAIRO Inform. Théor. Appl. **26**(3), 257–286 (1992)
21. Courcelle, B.: The monadic second-order logic of graphs. I. Recognizable sets of finite graphs. Information and Computation **85**(1), 12–75 (1990)
22. Courcelle, B.: The monadic second-order logic of graphs V: on closing the gap between definability and recognizability. Theoretical Computer Science **80**(2), 153 – 202 (1991)
23. Courcelle, B.: The expression of graph properties and graph transformations in monadic second-order logic. Handbook of Graph Grammars pp. 313–400 (1997)
24. Courcelle, B., Engelfriet, J.: Graph Structure and Monadic Second-Order Logic - A Language-Theoretic Approach, Encyclopedia of mathematics and its applications, vol. 138. Cambridge University Press (2012), http://www.cambridge.org/fr/knowledge/isbn/item5758776/?site_locale=fr_FR
25. Cygan, M., Fomin, F.V., Kowalik, L., Lokshtanov, D., Marx, D., Pilipczuk, M., Pilipczuk, M., Saurabh, S.: Parameterized Algorithms. Springer (2015)
26. Dawar, A., Grohe, M., Kreutzer, S.: Locally excluding a minor. In: LICS 2007. pp. 270–279. IEEE Computer Society (2007)
27. Dell, H., Van Melkebeek, D.: Satisfiability allows no nontrivial sparsification unless the polynomial-time hierarchy collapses. J. ACM **61**(4), 23:1–23:27 (Jul 2014). https://doi.org/10.1145/2629620, http://doi.acm.org/10.1145/2629620
28. Demaine, E., Hajiaghayi, M.: The bidimensionality theory and its algorithmic applications. The Computer Journal **51**(3), 292–302 (2007)
29. Demaine, E.D., Fomin, F.V., Hajiaghayi, M., Thilikos, D.M.: Bidimensional parameters and local treewidth. SIAM J. Discrete Math. **18**(3), 501–511 (2004)
30. Demaine, E.D., Fomin, F.V., Hajiaghayi, M., Thilikos, D.M.: Subexponential parameterized algorithms on graphs of bounded genus and H-minor-free graphs. Journal of the ACM **52**(6), 866–893 (2005)

31. Demaine, E.D., Hajiaghayi, M.T.: Linearity of grid minors in treewidth with applications through bidimensionality. Combinatorica **28**(1), 19–36 (2008)
32. Demaine, E.D., Hajiaghayi, M.: Bidimensionality: new connections between FPT algorithms and PTASs. In: Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2005). pp. 590–601. ACM-SIAM, New York (2005)
33. Demaine, E.D., Hajiaghayi, M.: Bidimensionality. In: Kao, M.Y. (ed.) Encyclopedia of Algorithms. Springer (2008)
34. Demaine, E.D., Hajiaghayi, M., Thilikos, D.M.: The bidimensional theory of bounded-genus graphs. SIAM J. Discrete Math. **20**(2), 357–371 (2006). https://doi.org/10.1137/040616929, https://doi.org/10.1137/040616929
35. Downey, R.G., Fellows, M.R.: Parameterized complexity. Springer-Verlag, New York (1999)
36. Downey, R., Fellows, M.: Fixed-parameter tractability and completeness. III. Some structural aspects of the W hierarchy. In: Complexity theory, pp. 191–225. Cambridge Univ. Press, Cambridge (1993)
37. Downey, R.G., Fellows, M.R.: Fixed-parameter tractability and completeness. I. Basic results. SIAM J. Comput. **24**(4), 873–921 (1995)
38. Downey, R.G., Fellows, M.R.: Fixed-parameter tractability and completeness II: On completeness for $W[1]$. Theoretical Computer Science **141**(1-2), 109–131 (1995)
39. Downey, R.G., Fellows, M.R., Prieto-Rodriguez, E., Rosamond, F.A.: Fixed-parameter tractability and completeness V: parametric miniatures (2003), manuscript
40. Downey, R.G., Fellows, M.R.: Fundamentals of Parameterized Complexity. Texts in Computer Science, Springer (2013). https://doi.org/10.1007/978-1-4471-5559-1, http://dx.doi.org/10.1007/978-1-4471-5559-1
41. Downey, R.G., Fellows, M.R., Langston, M.A.: The computer journal special issue on parameterized complexity: Foreword by the guest editors. The Computer Journal **51**(1), 1–6 (2008). https://doi.org/10.1093/comjnl/bxm111, http://comjnl.oxfordjournals.org/content/51/1/1.short
42. Eiben, E., Ganian, R., Szeider, S.: Meta-kernelization using well-structured modulators. Discrete Applied Mathematics **248**, 153 – 167 (2018). https://doi.org/https://doi.org/10.1016/j.dam.2017.09.018, http://www.sciencedirect.com/science/article/pii/S0166218X17304419, seventh Workshop on Graph Classes,Optimization, and Width Parameters, Aussois, France, October 2015
43. Fellows, M.R., A., L.M.: An analogue of the myhill-nerode theorem and its use in computing finite-basis characterisations (extended abstract). In: 30th Annual IEEE Symposium on Foundations of Computer Science, FOCS 1989, pp. 520–525. IEEE (1989)
44. de Fluiter, B.: Algorithms for Graphs of Small Treewidth. Ph.D. thesis, Dept. Computer Science, Utrecht University (1997)
45. Flum, J., Grohe, M.: Parameterized Complexity Theory. Texts in Theoretical Computer Science. An EATCS Series, Springer-Verlag, Berlin (2006)
46. Fomin, F.V., Demaine, E.D., Hajiaghayi, M.T., Thilikos, D.M.: Bidimensionality. In: Encyclopedia of Algorithms, pp. 203–207. Springer (2016). https://doi.org/10.1007/978-1-4939-2864-4_47, https://doi.org/10.1007/978-1-4939-2864-4_47
47. Fomin, F.V., Golovach, P., Thilikos, D.M.: Contraction bidimensionality: the accurate picture. In: 17th Annual European Symposium on Algorithms (ESA 2009), pp. 706–717. LNCS, Springer (2009)

48. Fomin, F.V., Golovach, P.A., Thilikos, D.M.: Contraction obstructions for treewidth. J. Comb. Theory, Ser. B **101**(5), 302–314 (2011)
49. Fomin, F.V., Lokshtanov, D., Misra, N., Ramanujan, M.S., Saurabh, S.: Solving d-SAT via backdoors to small treewidth. In: Indyk, P. (ed.) Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015. pp. 630–641. SIAM (2015). https://doi.org/10.1137/1.9781611973730.43, https://doi.org/10.1137/1.9781611973730.43
50. Fomin, F.V., Lokshtanov, D., Misra, N., Saurabh, S.: Planar F-deletion: Approximation, kernelization and optimal FPT algorithms. In: 53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012. pp. 470–479 (2012)
51. Fomin, F.V., Lokshtanov, D., Raman, V., Saurabh, S.: Bidimensionality and EPTAS. In: Randall, D. (ed.) Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011, San Francisco, California, USA, January 23-25, 2011. pp. 748–759. SIAM (2011). https://doi.org/10.1137/1.9781611973082.59, https://doi.org/10.1137/1.9781611973082.59
52. Fomin, F.V., Lokshtanov, D., Saurabh, S.: Bidimensionality and geometric graphs. In: Proceedings of the Twenty-third Annual ACM-SIAM Symposium on Discrete Algorithms. pp. 1563–1575. SODA '12, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA (2012), http://dl.acm.org/citation.cfm?id=2095116.2095240
53. Fomin, F.V., Lokshtanov, D., Saurabh, S.: Excluded grid minors and efficient polynomial-time approximation schemes. J. ACM **65**(2), 10:1–10:44 (2018). https://doi.org/10.1145/3154833, https://doi.org/10.1145/3154833
54. Fomin, F.V., Lokshtanov, D., Saurabh, S., Thilikos, D.M.: Bidimensionality and kernels. In: 21st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2010), pp. 503–510. ACM-SIAM (2010)
55. Fomin, F.V., Lokshtanov, D., Saurabh, S., Thilikos, D.M.: Bidimensionality and kernels. CoRR **abs/1606.05689** (2016), http://arxiv.org/abs/1606.05689, revised version
56. Fomin, F.V., Lokshtanov, D., Saurabh, S., Thilikos, D.M.: Kernels for (connected) dominating set on graphs with excluded topological minors. ACM Trans. Algorithms **14**(1), 6:1–6:31 (2018). https://doi.org/10.1145/3155298, https://doi.org/10.1145/3155298
57. Fomin, F.V., Lokshtanov, D., Saurabh, S., Zehavi, M.: Kernelization: Theory of Parameterized Preprocessing. Cambridge University Press (2019). https://doi.org/10.1017/9781107415157
58. Frick, M., Grohe, M.: Deciding first-order properties of locally tree-decomposable structures. J. Assoc. Comput. Mach. **48**(6), 1184–1206 (2001)
59. Gajarský, J., Hliněný, P., Obdržálek, J., Ordyniak, S., Reidl, F., Rossmanith, P., Villaamil, F.S., Sikdar, S.: Kernelization using structural parameters on sparse graph classes. Journal of Computer and System Sciences **84**, 219 – 242 (2017). https://doi.org/https://doi.org/10.1016/j.jcss.2016.09.002, http://www.sciencedirect.com/science/article/pii/S0022000016300812
60. Ganian, R., Slivovsky, F., Szeider, S.: Meta-kernelization with structural parameters. J. Comput. Syst. Sci. **82**(2), 333–346 (2016). https://doi.org/10.1016/j.jcss.2015.08.003, https://doi.org/10.1016/j.jcss.2015.08.003

61. Garnero, V., Paul, C., Sau, I., Thilikos, D.M.: Explicit linear kernels via dynamic programming. SIAM J. Discrete Math. **29**(4), 1864–1894 (2015)
62. Garnero, V., Sau, I., Thilikos, D.M.: A linear kernel for planar red-blue dominating set. Discrete Applied Mathematics **217**, 536–547 (2017). https://doi.org/10.1016/j.dam.2016.09.045, https://doi.org/10.1016/j.dam.2016.09.045
63. Giannopoulou, A.C., Pilipczuk, M., Raymond, J., Thilikos, D.M., Wrochna, M.: Linear kernels for edge deletion problems to immersion-closed graph classes. In: 44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland. pp. 57:1–57:15 (2017)
64. Grigoriev, A., Koutsonas, A., Thilikos, D.M.: Bidimensionality of geometric intersection graphs. In: Geffert, V., Preneel, B., Rovan, B., Štuller, J., Tjoa, A.M. (eds.) SOFSEM 2014: Theory and Practice of Computer Science: 40th International Conference on Current Trends in Theory and Practice of Computer Science, Nový Smokovec, Slovakia, January 26-29, 2014, Proceedings. pp. 293–305. Springer International Publishing (2014)
65. Grohe, M.: Logic, Graphs, and Algorithms, chap. in J.Flum, E.Grädel, T.Wilke (Eds), Logic and Automata — History and Perspectives, pp. 357 – 422. Amsterdam University Press (2007)
66. Grohe, M., Kawarabayashi, K., Marx, D., Wollan, P.: Finding topological subgraphs is fixed-parameter tractable. In: Fortnow, L., Vadhan, S.P. (eds.) Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6-8 June 2011. pp. 479–488. ACM (2011). https://doi.org/10.1145/1993636.1993700, https://doi.org/10.1145/1993636.1993700
67. Grohe, M., Kreutzer, S.: Methods for algorithmic meta theorems, chap. Model Theoretic Methods in Finite Combinatorics, pp. 181 – 206. Contemporary Mathematics (2011)
68. Gu, Q., Tamaki, H.: Improved bounds on the planar branchwidth with respect to the largest grid minor size. Algorithmica **64**(3), 416–453 (2012)
69. Guo, J., Niedermeier, R.: Linear problem kernels for NP-hard problems on planar graphs. In: Automata, languages and programming, LNCS, vol. 4596, pp. 375–386. Springer, Berlin (2007). https://doi.org/10.1007/978-3-540-73420-8_34, http://dx.doi.org/10.1007/978-3-540-73420-8_34
70. Guo, J., Niedermeier, R., Wernicke, S.: Fixed-parameter tractability results for full-degree spanning tree and its dual. In: Second International Workshop on Parameterized and Exact Computation (IWPEC), LNCS, vol. 4169, pp. 203–214. Springer (2006)
71. Gutin, G.Z.: Kernelization, Constraint Satisfaction Problems Parameterized above Average, pp. 1011–1013. Springer (2016). https://doi.org/10.1007/978-1-4939-2864-4_524, https://doi.org/10.1007/978-1-4939-2864-4_524
72. Gutin, G.Z., Yeo, A.: Constraint satisfaction problems parameterized above or below tight bounds: A survey. In: Bodlaender, H.L., Downey, R., Fomin, F.V., Marx, D. (eds.) The Multivariate Algorithmic Revolution and Beyond - Essays Dedicated to Michael R. Fellows on the Occasion of His 60th Birthday. Lecture Notes in Computer Science, vol. 7370, pp. 257–286. Springer (2012). https://doi.org/10.1007/978-3-642-30891-8_14, https://doi.org/10.1007/978-3-642-30891-8_14
73. Hajiaghayi, M.T.: The bidimensionality theory and its algorithmic applications. Ph.D. thesis, Department of Mathematics, Massachusetts Institute of Technology. (2005)

74. Jansen, B.M.P., Kratsch, S.: A structural approach to kernels for ilps: Treewidth and total unimodularity. In: Bansal, N., Finocchi, I. (eds.) Algorithms - ESA 2015 - 23rd Annual European Symposium, Patras, Greece, September 14-16, 2015, Proceedings. Lecture Notes in Computer Science, vol. 9294, pp. 779–791. Springer (2015). https://doi.org/10.1007/978-3-662-48350-3_65, https://doi.org/10.1007/978-3-662-48350-3_65
75. Kanj, I.A., Pelsmajer, M.J., Xia, G., Schaefer, M.: On the induced matching problem. In: Proceedings of the 25th Annual Symposium on Theoretical Aspects of Computer Science (STACS 2008), vol. 08001, pp. 397–408. Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany, Berlin (2008)
76. Kim, E.J., Langer, A., Paul, C., Reidl, F., Rossmanith, P., Sau, I., Sikdar, S.: Linear kernels and single-exponential algorithms via protrusion decompositions. ACM Trans. Algorithms **12**(2), 21:1–21:41 (2016)
77. Kim, E.J., Serna, M.J., Thilikos, D.M.: Data-compression for parametrized counting problems on sparse graphs. In: Hsu, W., Lee, D., Liao, C. (eds.) 29th International Symposium on Algorithms and Computation, ISAAC 2018, December 16-19, 2018, Jiaoxi, Yilan, Taiwan. LIPIcs, vol. 123, pp. 20:1–20:13. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik (2018). https://doi.org/10.4230/LIPIcs.ISAAC.2018.20, https://doi.org/10.4230/LIPIcs.ISAAC.2018.20
78. Kloks, T.: Treewidth, Computations and Approximations, Lecture Notes in Computer Science, vol. 842. Springer (1994)
79. Kreutzer, S.: Algorithmic meta-theorems. Electronic Colloquium on Computational Complexity (ECCC) **16**, 147 (2009)
80. Lokshtanov, D., Mnich, M., Saurabh: Linear kernel for planar connected dominating set. In: Proceedings of Theory and Applications of Models of Computation, (TAMC 2009). Lecture Notes in Comput. Sci., Springer (2009)
81. Misra, N., Raman, V., Saurabh, S.: Lower bounds on kernelization. Discrete Optimization **8**(1), 110 – 128 (2011). https://doi.org/https://doi.org/10.1016/j.disopt.2010.10.001, http://www.sciencedirect.com/science/article/pii/S157252861000068X, parameterized Complexity of Discrete Optimization
82. Moser, H., Sikdar, S.: The parameterized complexity of the induced matching problem in planar graphs. In: Proceedings First Annual International WorkshopFrontiers in Algorithmics (FAW), LNCS, vol. 4613, pp. 325–336. Springer (2007)
83. Niedermeier, R.: Invitation to Fixed-Parameter Algorithms. Oxford University Press (2006). https://doi.org/10.1093/ACPROF:OSO/9780198566076.001.0001, https://doi.org/10.1093/ACPROF:OSO/9780198566076.001.0001
84. Robertson, N., Seymour, P.D.: Graph Minors. XX. Wagner's conjecture. Journal of Combinatorial Theory, Series B **92**(2), 325–357 (2004)
85. Thilikos, D.M.: Graph minors and parameterized algorithm design. In: The Multivariate Algorithmic Revolution and Beyond - Essays Dedicated to Michael R. Fellows on the Occasion of His 60th Birthday. pp. 228–256 (2012)
86. Thilikos, D.M.: Bidimensionality and parameterized algorithms (invited talk). In: 10th International Symposium on Parameterized and Exact Computation, IPEC 2015, September 16-18, 2015, Patras, Greece. pp. 1–16 (2015)
87. Zoros, D.: Obstructions and Algorithms for Graph Layout Problems. Ph.D. thesis, National and Kapodistrian University of Athens, Department of Mathematics (July 2017)