Towards Fully Multivariate Algorithmics: Parameter Ecology and the Deconstruction of Computational Complexity[☆,☆☆]

Michael R. Fellows[a,*], Bart M. P. Jansen[b], Frances Rosamond[a]

[a] *School of Engineering and Information Technology, Charles Darwin University, Australia*
[b] *Dept. of Information and Computing Sciences, Utrecht University, The Netherlands*

Abstract

The aim of this article is to motivate and describe the parameter ecology program, which studies how different parameters contribute to the difficulty of classical problems. We call for a new type of *race* in parameterized analysis, with the purpose of uncovering the boundaries of tractability by finding the smallest possible parameterizations which admit FPT-algorithms or polynomial kernels. An extensive overview of recent advances on this front is presented for the Vertex Cover problem. Moving even beyond the parameter ecology program we advocate the principle of model enrichment, which raises the challenge of generalizing positive results to problem definitions with greater modeling power. The computational intractability which inevitably emerges can be deconstructed by introducing additional parameters, leading towards a theory of fully multivariate algorithmics.

Keywords: parameter ecology, multivariate algorithmics, parameterized complexity
2010 MSC: 68Q17 68Q25 68W40 05C85

1. Introduction

Parameterized complexity starts from the premise that there are usually secondary measurements, apart from the primary measurement of overall input

[☆]This work was supported by the Netherlands Organization for Scientific Research (NWO), project "KERNELS: Combinatorial Analysis of Data Reduction".
[☆☆]A preliminary version of this work appeared in the proceedings of the 20th International Workshop on Combinatorial Algorithms (IWOCA 2009). This is the author-accepted manuscript of a paper originally published in the European Journal of Combinatorics (Volume 34, Issue 3, April 2013, Pages 541–566). The final publication is available at link.springer.com through `http://dx.doi.org/10.1016/j.ejc.2012.04.008`.
[*]Corresponding author
Email addresses: `Michael.Fellows@cdu.edu.au` (Michael R. Fellows), `B.M.P.Jansen@uu.nl` (Bart M. P. Jansen), `Frances.Rosamond@cdu.edu.au` (Frances Rosamond)

size, that can significantly affect the computational complexity of a problem, in qualitatively different ways that merit systematic investigation. Parameterized complexity makes room for one such measurement, the *parameter*, and turns on the contrast between the classes of bivariate functions $\mathcal{FPT}$ and $\mathcal{XP}$.

This is formalized by saying that a *parameterized problem* Π takes as input a pair (x, k), where x is a string over a finite alphabet and k is the *parameter* (usually a positive integer). The problem Π is *fixed-parameter tractable* (FPT) if it can be solved in time $f(k)n^c$ where n is the overall input size, that is, $n = |(x, k)|$, c is a constant, and $f(k)$ is some function of the parameter k. $\mathcal{FPT}$ denotes the class of bivariate functions of this form, and FPT denotes the class of parameterized problems solvable in FPT-time. $\mathcal{XP}$ is the class of bivariate functions of the form $\mathcal{O}(n^{g(k)})$, and XP denotes the class of parameterized problems that can be decided in time bounded by a function in $\mathcal{XP}$.

Viewing complexity theories as concretely driven by contrasting function classes, the classical framework of P *versus* NP is about the contrast between the univariate function classes: $\mathcal{P}$, solvability in time $\mathcal{O}(n^c)$, and functions of the form $\mathcal{O}(2^{n^c})$. In practical algorithmics one would much prefer an algorithm with a runtime bounded by a function (of the overall input size) in $\mathcal{P}$. Hardness for NP indicates probably being stuck with a running time of the latter kind.

FPT generalizes P; parameterized complexity theory can be viewed as a natural two-dimensional generalization of the one-dimensional classical framework. From a theoretical point of view, parameterized analysis aims to obtain a better understanding of the structure of a problem: how do different parameters contribute to the difficulty of an instance, in terms of the function classes $f(k)n^c$ and $n^{g(k)}$? But there is also a very practical reason to study the parameterized complexity of hard problems, because the parameterized toolkit has the potential to yield two types of *deliverables* that facilitate an attack on the seeming intractability of NP-hard problems. Historically the most prominent deliverable takes the form of an algorithm to solve the problem in FPT-time, which is relevant for parameterizations which are small on real-world instances. But there is another desired outcome of parameterized analysis, a kernelization algorithm, and there have been many exciting developments in this subfield over the last several years [9, 70]. Whereas FPT is (in general) concerned with exponential-time algorithms, a *kernelization algorithm* (or *kernel*) for a parameterized problem Π is an algorithm which transforms an instance (x, k) in *polynomial* $\mathcal{O}(n^c)$ time into an instance (x', k') such that $(x, k) \in \Pi \Leftrightarrow (x', k') \in \Pi$, with a guarantee on the size of the output instance: $|x'|, k' \leq g(k)$ for some function g, which is called the *size* of the kernel. Since many kernelization algorithms take the form of a set of reduction rules, they are widely applicable as preprocessing routines, and model the common practice of applying data-reduction heuristics before attacking computationally difficult problems.

It turns out that, in general, kernelization is just the study of parameterized algorithms in disguise: a decidable parameterized problem has a kernel if and only if it is FPT [46, Lemma 4.8]. But the search for a problem kernel becomes interesting (and, one could argue, useful) once we demand that the function g which bounds the size of the reduced instance is polynomial. The quest for

such polynomial kernels began in earnest after Alber et al. [2] found a kernel with $335k$ vertices for k-Planar Dominating Set in 2003, and has spawned a myriad of interesting theoretical and practical results; we will see some of them later in this survey.

The two types of deliverables motivate two types of "races" in parameterized complexity research: the race for the smallest possible function $f(k)$ in the running time of an exact algorithm, and the race for the smallest possible function $g(k)$ to upper bound the size of a kernel. These races are well-established, and the current leader boards are exhibited on the FPT community wiki [102]. We will see later that the parameter ecology program gives rise to a new kind of race, but first we continue our introduction into parameterized complexity. Having defined the two types of good news that parameterized analysis potentially has to offer, let us give two classical examples which also illustrate the expressibility of negative results in this theory.

k-Vertex Cover
Instance: An undirected graph G and integer k.
Parameter: k.
Question: Is there a set $S \subseteq V(G)$ of size at most k which contains at least one endpoint of every edge?

k-Dominating Set
Instance: An undirected graph G and integer k.
Parameter: k.
Question: Is there a set $S \subseteq V(G)$ of size at most k such that each vertex not in S, has a neighbor in S?

Although the classical versions of both problems are *NP*-complete, the input *parameter* k contributes to the complexity of these two problems in two qualitatively different ways.

1. There is a simple *bounded search tree* algorithm for k-Vertex Cover that runs in time $\mathcal{O}(2^k n)$ [43].
2. The best known algorithm for k-Dominating Set uses fast matrix multiplication [48], and gives a running time of $\Theta(n^{\Theta(k)})$ — it is only a minor improvement on the brute force algorithm of trying all k-subsets, which takes $\mathcal{O}(n^{k+1})$ time for a graph with n vertices.

Table 1 shows the contrast between these two kinds of complexity. Does this mean that Vertex Cover is an easier problem than Dominating Set? The difference in running times certainly seems to suggest that this is the case, but the answer to this question is more subtle; we will discuss it at length in Section 2.

Classically, evidence that a problem is unlikely to have an algorithm with a runtime in the good class $\mathcal{P}$ is given by determining that it is *NP*-hard, *PSPACE*-hard, *EXP*-hard, etc. In parameterized complexity analysis there are analogous means to show likely parameterized intractability. The current tower

	$n = 50$	$n = 100$	$n = 150$
$k = 2$	625	2,500	5,625
$k = 3$	15,625	125,000	421,875
$k = 5$	390,625	6,250,000	31,640,625
$k = 10$	1.9×10^{12}	9.8×10^{14}	3.7×10^{16}
$k = 20$	1.8×10^{26}	9.5×10^{31}	2.1×10^{35}

Table 1: The Ratio $\frac{n^{k+1}}{2^k n}$ for Various Values of n and k.

of the main parameterized complexity classes is:

$$FPT \subseteq M[1] \subseteq W[1] \subseteq M[2] \subseteq W[2] \subseteq \cdots \subseteq W[P] \subseteq XP.$$

The cited running time shows that Vertex Cover lies in FPT, parameterized by the size k of a solution. The familiar k-Independent Set problem is complete for $W[1]$ and k-Dominating Set is complete for $W[2]$ (cf. [43]). The naturally parameterized Bandwidth problem is hard for $W[t]$ for all t [11], but not known to belong to $W[P]$. The best known algorithms for the parameterized k-Independent Set and k-Dominating Set problems run in $\Omega(n^{\Omega(k)})$ time [48, 97]. The parameterized class $W[1]$ is strongly analogous to NP, because the k-Step Halting Problem for Turing machines of unlimited nondeterminism and alphabet size is complete for $W[1]$ (cf. [42]). FPT is equal to $M[1]$ if and only if the so-called *Exponential Time Hypothesis* fails [21, 44, 75].

Chen et al. [22] proved that if there is an algorithm for the k-Independent Set problem that runs in time $\mathcal{O}(n^{o(k)})$ then FPT is equal to $M[1]$. From this it follows that if there is an algorithm for the k-Dominating Set problem that runs in time $\mathcal{O}(n^{o(k)})$ then FPT is equal to $M[1]$.

The parameterized complexity class $W[1]$ is entirely natural, being anchored (completely) by a natural form of the Halting Problem. The parameterized complexity class $M[1]$ also arises naturally, through a kind of *complexity renormalization*. The parameterized Vertex Cover problem can be solved in time $\mathcal{O}(2^k n)$ by the elementary search tree algorithm $\mathcal{A}$. Consider then the following parameterized problem aimed at extending this FPT success:

> Renormalized k-Vertex Cover
> **Instance:** An undirected graph G and integer k.
> **Parameter:** k.
> **Question:** Is there a set $S \subseteq V(G)$ of size at most $k \log n$ which contains at least one endpoint of every edge?

Note that algorithm $\mathcal{A}$ allows us to solve this problem in time $\mathcal{O}(n^k)$, and it is therefore quite natural to ask if this more ambitious (renormalized) version of Vertex Cover might be FPT or not. One way of defining the increasingly important parameterized complexity class $M[1]$ is as simply the class of all those parameterized problems that are FPT-equivalent to Renormalized k-Vertex

Cover. The definition of $M[t]$ for $t \geq 2$ is elaborated in the monograph of Flum and Grohe [59].

The results of Chen et al. [22] mentioned above give some perspective on how hard it might be to significantly improve on brute force for the parameterized k-Independent Set and k-Dominating Set problems (if this is possible). Because it is known that FPT is a proper subset of XP [43, Chapter 15], one cannot solve k-Dominating Set in time $\mathcal{O}(n^{o(k)})$ unless $W[1] \neq XP$.[1]

For further background on parameterized complexity we refer the reader to the textbooks [43, 59, 99], the double special issue of surveys of aspects of the field and various application areas [45], and to the recent survey by Downey and Thilikos [47]. Having given an overview of the positive and negative results possible in the theory of parameterized complexity, let us move in the direction of the ecology program by reconsidering our two favorite graph problems. For k-Vertex Cover and k-Dominating Set, the parameter is the size of the solution being sought. Indeed, when studying the decision version of an optimization problem, the most *natural* parameter is often the desired solution size. But a parameter that affects the complexity of a problem can be many things, as we shall see when we consider the problem ML Type Checking in the next section.

2. The Complexity Ecology of Parameters

2.1. Why Ecology?

ML is a logic-based programming language for which relatively efficient compilers exist. One of the problems the compiler must solve is the checking of the compatibility of type declarations. This problem is known to be complete for EXP (deterministic exponential time), so the situation appears discouraging from the standpoint of classical complexity theory [74]. However, the implementations work well in practice because the ML Type Checking problem is FPT with a running time of $\mathcal{O}(2^k n)$, where n is the size of the program and k is the maximum nesting depth of the type declarations [89]. Despite the severe intractability of type checking from a classical point of view, the implemented ML compilers (that include type-checking subroutines) work efficiently because they *exploit the bounded nesting depth* of the inputs. The explanation is that human-composed programs typically have a maximum type-declaration nesting depth of $k \leq 5$. The reason that naturally occurring programs have small nesting depth is because the programs would otherwise risk becoming incomprehensible to the programmer creating them!

Type checking is not an optimization problem; hence there is no obvious "natural" parameter. The example shows that relevant secondary measurements that affect problem complexity can take *many different forms*, such as the size of the solution, aspects of the structure of typical instances, aspects of

[1] This is a statement in the parameterized world of something roughly like $NP \neq EXP$ in the classical world. See the full version of Williams' paper [117] for further discussion.

k \ Π	TW	BW	VC	DS	G	ML
TW	☺☹[7, 10]	☹☹	☺☹[10, 16]	☺☹[10, 16]	☺☹[10, 82]	☺☹[10, 16]
BW	☺☹[7, 10]	☹☹[11]	☺☹[10, 16]	☺☹[10, 16]	☺☹[82] ★	☺☹[10, 16]
VC	☺☺[15]	☺ ? [55]	☺☺[25, 96]	☺☹[16, 41]	☺ ? [82]	☺☹[16, 41]
DS	☹☹★	? ?	☹☹★	☹☹[43]	? ?	☹☹[43]
G	? ?	☹☹[67]	☹☹★	☹☹[67]	☺ ? [82]	☹☹[67]
ML	☺☺[84] ★	☺ ? [55]	☺☺[55]	☺☺[55]	☺☺[84] ★	☺☺[49]

Table 2: The complexity ecology of parameters. Each column corresponds to a classical problem Π, and each row to a parameterization k of that problem. An entry shows the current state of knowledge concerning the existence of an FPT algorithm or polynomial kernel. If the first symbol is ☺ then the parameterized problem is in *FPT*, while ☹ signifies hardness for $W[1]$. The second symbol indicates whether there is a polynomial kernel (☺) or not (☹). The kernelization lower-bounds for problems in *FPT* are conditioned on one of the following two assumptions: (a) $NP \not\subseteq coNP/\text{poly}$, or (b) that not all *coNP*-complete problems admit distillation algorithms (cf. [10]). Entries marked with ★ are not stated explicitly in the literature, but follow from known results or unpublished insights.

the algorithmic approach, or the quality of an approximation (cf. [100]). For any real-world problem, there may be several such secondary measurements in various ranges of magnitude that should be considered, or measurements with qualitatively different roles, such as structural (e.g., treewidth) and solution size.

What the example of ML Type Checking points to (we think) is that often the "inputs" to computational problems of interest to real-world algorithmics are not at all arbitrary, but rather are produced by other natural computational processes (e.g., the thinking processes and abilities of the programmer) that are themselves subject to computational complexity constraints. In this way, the natural input distributions encountered by abstractly defined computational problems often have inherited structural regularities and restrictions (relevant parameters, in the sense of parameterized complexity) due to the natural complexity constraints on the generative processes. This connection is what we refer to as the *ecology* of computation.

Having identified a reason to expect some kind of structure in our real-world problem instances, we are driven to explore how to exploit that structure in order to solve problems more efficiently. It therefore seems to be useful to know how all the various parameterized structural notions interact with the computational objectives one might have. For example, faced with the apparent parameterized intractability of k-Dominating Set, one could ask whether a bound on the treewidth of the input graphs allows Dominating Set to be solved more efficiently. It has long been known that this is the case: when a tree-decomposition of width k is supplied, a minimum size dominating set can be computed in $\mathcal{O}(3^k k^2 n)$ time [116]. The familiar paradigm of efficiently solving various problems for graphs of bounded treewidth just represents one row of a matrix of algorithmic questions.

Table 2 illustrates the idea. We use here the shorthand: TW is Treewidth, BW is Bandwidth, VC is Vertex Cover, DS is Dominating Set, G is

(orientable) Genus and ML is Maximum Number of Leaves in a Spanning Tree. An entry in the table describes the current state of knowledge about the parameterized complexity and the existence of polynomial kernels when the input graph is assumed to have a structural bound described by the *row*, and the problem described by the *column* is to be solved to optimality. It turns out that there are some non-trivial choices to be made when it comes to formalizing such non-standard parameterizations, but we defer the discussion of these matters to Section 2.2 and focus on the main ideas. The entry in the second row and fourth column of the table indicates that there is an *FPT* algorithm to optimally solve the Dominating Set problem for a graph G of bandwidth at most k, but that there is no polynomial kernel for this problem unless $NP \subseteq coNP/\text{poly}$. The entry in the fourth row and second column indicates that it is unknown whether Bandwidth can be solved optimally by an *FPT* algorithm when the parameter is a bound on the domination number of the input. The table just gives a few examples of the unbounded conceptual matrix that we are concerned with. The question marks in the table correspond to open problems; we highlight the most relevant of these.

Open Problem 1. Does the natural parameterization of the orientable Genus problem admit a polynomial kernel?

Open Problem 2. Does Bandwidth parameterized by vertex cover admit a polynomial kernel?

Open Problem 3. Is Treewidth in *XP*, or even in *FPT*, parameterized by the genus of the input graph?

Note that a positive answer to the last question would imply that the treewidth of planar graphs can be computed in polynomial time (as is known to be the case for the related width-measure branchwidth [111]).

Using the information in the table, we can come back to the question whether Dominating Set is harder to solve than Vertex Cover. Classically this is certainly not the case; both are *NP*-complete, hence solved in polynomial time on a non-deterministic machine, but presumably require exponential time to solve deterministically. We can use Table 2 to compare the difficulty of the two problems under various parameterizations. The two relevant columns suggest that when it comes to the existence of FPT-algorithms, both problems are equally hard — Vertex Cover and Dominating Set are *FPT* for exactly the same parameterizations. A careful reader will note that the existence of polynomial kernels differs, for the third row: Vertex Cover under its natural parameterization has a kernel with $2k$ vertices [25], while Dominating Set parameterized by vertex cover does not, unless $NP \subseteq coNP/\text{poly}$ [41]. So rather than saying that Dominating Set is harder than Vertex Cover, we can conclude that (for the considered parameterizations) both problems are equally difficult to *solve*, but Vertex Cover is easier to *kernelize*. Kernels for Vertex Cover will be discussed at length in Section 4. For now, we keep our attention on Table 2 but switch the perspective from columns to rows.

Observe that all parameterizations by vertex cover number are in FPT, while none of the parameterizations by domination number are known to be in FPT! So the difference in parameterized complexity status for k-Vertex Cover and k-Dominating Set is not caused by Dominating Set being harder than Vertex Cover; it is because the *parameterization* by domination number is harder than the *parameterization* by vertex cover number. This is explained by noting that graphs with a vertex cover of size k are more restricted than those with a dominating set of size k — the non-trivial connected graphs with a size-k vertex cover form a proper subset of those with a size-k dominating set — and therefore a restriction on the vertex cover number gives an algorithm "more structure" to exploit than a bound of the domination number. The take-away message from this comparison is: the parameterized complexity of a problem is determined by the *parameterization*, rather than the underlying *classical problem*[2]!

2.2. Formalizing Structural Parameterizations

Let us come back to the question of what it formally means to parameterize a problem by a structural measure, by returning to the example of Dominating Set on a graph of bandwidth at most k. If we were to give a full definition of this problem, it could be the following:

> Dominating Set parameterized by bandwidth
> **Instance:** An undirected graph G and integer q.
> **Parameter:** A value k which bounds the bandwidth of G.
> **Question:** Does G have a dominating set of size at most q?

The classification of this problem as FPT in Table 2 is based on the fact that the treewidth of a graph is not greater than its bandwidth [8], which allows the use of the algorithm to solve Dominating Set on graphs of bounded treewidth mentioned earlier. But this intuition about the difficulty of the problem is not captured by all possible formalizations, and this is partly due to issues which are hidden by the instance-parameter-problem notation. At the most elementary level, a parameterized problem is the set of pairs (x, k) which encode yes-instances to the problem. Which language Q corresponds to the Dominating Set problem parameterized by bandwidth, as we have just put it? A first answer might look like this:

$$Q = \big\{(x,k) \,\big|\, x \text{ encodes a graph } G \text{ of bandwidth at most } k \text{ and an integer } q, \text{ such that } G \text{ has a dominating set of size at most } q\big\}.$$

Unfortunately the behavior of this formalization differs radically from our intended meaning, which is best illustrated by noting that membership in Q is $W[t]$-hard to decide for all t! Since a graph G on n vertices trivially has a dominating set of size n, by our current definition the instance $(x := \langle G, n\rangle, k)$ is contained in Q if and only if G has bandwidth at most k, and this results in the

[2]Indeed, all decidable problems are FPT parameterized by the size of the input.

aforementioned complexity. The problem comes from the fact that the distinction between parameter and solution value was lost in the formalization, and was effectively turned into the task of simultaneously deciding the bandwidth and the domination number of the input graph. What the *right* formalization is, depends on the question that we intend to ask: do we want to study (I) how hard it is to *verify and exploit* a bounded parameter value, or (II) how to *exploit* structure when we are *promised* it is there but we do not *know* it yet, or (III) what we can do when a witness for the structure is *supplied along with the input*? Clearly, the given definition of the language Q corresponds to (I), the task of verifying a claimed bound on the input parameter and simultaneously solving the problem at hand. If it is $W[1]$-hard to verify the parameter value, then the resulting parameterizations generally do not lie in FPT.

Using a promise problem. The most straight-forward way out of our conundrum is to model the parameterization as a *promise problem*, which takes the form of *two* disjoint languages: one containing the YES-instances which satisfy the structural promise, and one containing the NO-instances satisfying the promise. An algorithm decides a promise problem if it accepts all YES-instances and rejects all NO-instances; there is no restriction on the behavior on inputs which are not contained in either. In our concrete example, the set Q defined earlier contains the YES-instances, and we define the set of well-formed NO-instances Q' as the pairs (x, k) for which x encodes a tuple $\langle G, q\rangle$ where G has bandwidth at most k, and G does *not* have a dominating set of size q. Since the instances where the bandwidth of G exceeds k fall outside the range of well-formed YES- or NO-instances, the restriction to a promise problem effectively means that we do not have to verify the claimed bound on the bandwidth of the input graph; all we demand is that *if* the bandwidth of the input graph is suitably bounded, *then* the algorithm gives the correct answer in FPT-time. The promise version of Dominating Set parameterized by bandwidth is indeed contained in FPT: on input (G, q) and the value k, the algorithm first computes a tree decomposition of width at most k in $f(k)n$ time using Bodlaender's algorithm [7], and then applies the algorithm of van Rooij et al. [116]. Observe that this algorithm takes too long on malformed instances where k does not bound the bandwidth of G, but the definition of a promise problem allows for this.

When we aim to study how hard it is to exploit a certain structure, a formalization as a promise problem allows us to avoid the issue of having to verify the presence of this structure; it is suitable when asking questions of type (II) mentioned above. Unfortunately, the use of a promise problem means that whenever we apply theorems from the FPT-toolkit which were proven for arbitrary decision problems, we have to ensure that they still hold in the promise setting. The question "can we exploit this *given* structure in the instance to solve the problem efficiently" leads us to an alternative that does not rely on promise problems.

Supplying a witness in the input. When studying a structural parameterization of problem Π, we can conceptually simplify matters by studying two separate

questions: (a) how hard is it to compute the relevant structural measure, either exactly or approximately, and (b) how hard is it to solve Π when a witness for the structure is given? The first problem is subject of study when considering the parameterized complexity or approximation complexity of the optimization problem corresponding to the structural measure, whereas the most interesting question from the viewpoint of parameter ecology is question (b). We could therefore opt for the following alternative definition of our running example:

> DS PARAMETERIZED BY THE BANDWIDTH OF A GIVEN LAYOUT
> **Instance:** An undirected graph G, an integer q, and a linear layout of G given by a bijection $f\colon V(G) \to \{1, \ldots, |V(G)|\}$.
> **Parameter:** The bandwidth $k := \max_{\{u,v\} \in E(G)} |f(i) - f(j)|$ of f.
> **Question:** Does G have a dominating set of size at most q?

The language of YES-instances corresponding to this definition contains those pairs (x, k) where x encodes a tuple $\langle G, q, f \rangle$ such that f is a linear layout of bandwidth at most k for G, and G has a dominating set of size at most q. By supplying a layout in the input, we can *test in polynomial time whether an instance obeys the claimed bound* on its bandwidth or not, and this idea generalizes to all parameterizations corresponding to problems in *NP*. By combining this test with the treewidth algorithms sketched earlier, we can classify this formalization as *FPT*.

Observe that since the bandwidth of the *given* layout is used as the parameter, rather than the optimum value, this approach is generally only suitable only for minimization parameters, i.e., parameters where smaller means better. For an example of why this approach may not result in relevant questions for maximization parameters, consider the DOMINATING SET problem parameterized by the number of leaves in a given spanning tree of the graph:

> DS PARAMETERIZED BY # OF LEAVES IN A GIVEN SPANNING TREE
> **Instance:** Graph G, an integer q, and a spanning tree T of G.
> **Parameter:** The number of leaves k in tree T.
> **Question:** Does G have a dominating set of size at most q?

It is an easy exercise to devise a polynomial-time transformation that given a graph G, outputs a graph G' and a Hamiltonian path in G' such that the domination number of G' is exactly one larger than that of G. Using this transformation, we can reduce the classical DOMINATING SET problem to our last parameterization of DOMINATING SET with a *constant* parameter value $k = 2$, by using the Hamiltonian path as a spanning tree with exactly two leaves. Hence the considered parameterized problem is already *NP*-complete for constant parameter values, and is therefore not interesting. The reason is of course that when parameterizing by the quality of a given witness to the structural measure, having a suboptimal witness for minimization parameters gives us a *larger* parameter (allowing us more running time and a larger output for a kernelization algorithm), whereas for maximization parameters having a suboptimal witness demands a faster output and smaller kernel. So when we are in pursuit of objective (III), studying what we can do when the structure is *supplied along with the*

input, we can generally avoid the issue of a promise problem for minimization problems; for maximization parameters we nevertheless have to resort to them.

In summary, we have seen that there are various ways to formalize a structural parameterization, depending on which type of question (I)-(III) you intend to answer. Each way of formalizing comes at a price. One should be informed of these issues when studying the ecology of parameters, to choose the formalization which best suits the research interest. Various choices have been made in the literature. In the study of FPT algorithms on graphs of bounded treewidth, one often assumes that a tree decomposition is given along with the input. Although Treewidth is *FPT* (and therefore a tree decomposition can be computed in FPT-time, if none is given), computing an optimal decomposition currently takes $\mathcal{O}(2^{\mathcal{O}(k^3)}n)$ time [7, 103], and this would be the bottleneck for many linear-time algorithms. When studying kernelization complexity under structural minimization-parameters, the model where the structural description is given along with the input is often used (e.g., [34, 77]) whereas the setting of a promise problem was the implicit foundation for work on parameterizations by max leaf number [14, 55].

Open problem: Structure in the input or not. The type of considerations made in this section are the crucial issue behind our next open problem. For a graph class $\mathcal{F}$ and a graph G, we say that $S \subseteq V(G)$ is a vertex-deletion set to $\mathcal{F}$ if $G - S$, the graph obtained from G by deleting all vertices in S and their incident edges, is contained in $\mathcal{F}$. The following proposition gives the necessary background for the open problem.

Proposition 1 (Cf. [20])**.** *Given a graph G and a size-k vertex-deletion set $S \subseteq V(G)$ to a perfect graph, a maximum independent set in G can be computed in $\mathcal{O}(2^k n^c)$ time for some constant c.*

Proof. We exploit the fact that Independent Set is polynomial-time solvable on perfect graphs, and try all ways in which an independent set could intersect S. For each *independent* subset $S' \subseteq S$, the largest independent sets I in G with $I \cap S = S'$ are exactly the sets $S' \cup I'$ for which I' is a maximum independent set in $G - (S \cup N_G(S'))$: observe that I' cannot contain any vertices of S, nor any vertices of $N_G(S')$ (else it would not be independent). Since $G - (S \cup N_G(S'))$ is an induced subgraph of the perfect graph $G - S$, and perfect graphs are hereditary, a maximum independent set of the perfect graph $G - (S \cup N_G(S'))$ can be computed in polynomial time by the algorithm of Grötschel, Lovász and Schrijver [69, Chapter 9]. Hence we find a maximum independent set in G as the largest set taken over all choices of S'. □

The proposition shows that Independent Set is in *FPT* parameterized by the size of a *given* vertex-deletion set to a perfect graph. It is not at all clear whether this is also the case when S is not given in the input. If it would be possible to compute an $f(\text{OPT})$-approximation to the smallest perfect-deletion set in FPT-time, then combining the approximation algorithm with Proposition 1 would give an FPT-algorithm for the parameterization by a promised bound

on the vertex-deletion distance to a perfect graph. But whether or not such an approximation algorithm exists is unclear; it is $W[2]$-hard to compute the minimum vertex-deletion distance to a perfect graph [73], and the close connection between the Perfect Deletion problem and Hitting Set exhibited in that paper suggests that Perfect Deletion may be as hard to approximate in FPT-time as Dominating Set. The parameterized approximation complexity of the latter problem is still a big open problem in the field of parameterized approximation [91]. Thus we arrive at our open problem.

Open Problem 4. Is the Independent Set problem (or equivalently, Vertex Cover) in *FPT* when parameterized by a (promised) bound on the vertex-deletion distance to a perfect graph, without giving the deletion set in the input?

A similar type of question can be asked for 3-Coloring, which was shown to be *FPT* parameterized by the size of a *given* dominating set [78].

Open Problem 5. Is 3-Coloring in *FPT* when parameterized by a promised bound on the domination number of the graph?

2.3. Systematically Attacking a Row

Exploration of the parameter ecology table is facilitated by noting that there are various techniques which can be used to systematically attack the entries of a row — or in other words, that there are techniques which appear to be widely applicable to various different problems under the same parameterization. These tools may help to quickly classify a parameterized problem as *FPT*, or as having a polynomial kernel. In this section we explore some of these techniques. After the initial classification, *the game begins* to find better FPT algorithms and smaller kernels. For many problems in *FPT*, remarkable improvements are eventually achieved [102].

2.3.1. Courcelle's Theorem for Graphs of Bounded Treewidth

The study of problems on graphs of bounded treewidth is well-developed and supports many systematic approaches based on automata-theory and logic, the most famous example being Courcelle's theorem [3, 7, 16, 31, 43, 59, 99]. Formulating a problem in Monadic Second-Order Logic is a quick way of showing that a problem is *FPT* on graphs of bounded treewidth. As knowledge of these techniques is widespread, let us just mention a recent breakthrough result in this area. Cygan et al. [35] have shown that the solvability in single-exponential $2^{\mathcal{O}(k)}n^c$ time (for a given tree decomposition of width k) which was known for many problems whose solutions can *locally* be verified, can also be extended to *connectivity* problems at the cost of using randomization. For a multitude of problems, the phenomenon of solvability in randomized single-exponential time on graphs of bounded treewidth has since been captured by a special kind of logic dubbed Existential Counting Modal Logic [104].

2.3.2. Well-Quasi-Ordering

One of the most powerful *FPT* classification tools is *well-quasi-ordering* (WQO). For completeness we briefly recall some of the main notions of WQO-theory; the interested reader is referred to the textbook [43, Chapter 7] for more details. A *quasi-ordering* on an (infinite) set S is a reflexive and transitive relation $\leq_S$ on S. A quasi-ordered set $\langle S, \leq_S \rangle$ is a *well-quasi-order* if for all infinite sequences $A = a_1, a_2, \ldots$ over S there are indices $i < j$ such that $a_i \leq_S a_j$ and $a_j \not\leq_S a_i$. A *lower ideal* in a WQO is a subset $S' \subseteq S$ which is closed downwards under $\leq_S$: if $x \in S'$ and $y \leq_S x$, then $y \in S'$.

The main algorithmic power of WQO comes from the fact that every lower ideal is characterized by a finite obstruction set: for every ideal $S' \subseteq S$ there is a *finite* set of minimal forbidden elements $O_{S'}$ such that $x \in S' \Leftrightarrow \neg\exists y \in O_{S'} : y \leq_S x$. If order testing under $\leq_S$ is tractable, then the existence of a finite obstruction set leads to a tractable algorithm for deciding membership of an element x in a lower ideal S': simply test for every element y in the obstruction set $O_{S'}$ whether $y \leq_S x$.

The classic example in this area is of course the celebrated Graph Minor Theorem by Robertson and Seymour, which says that finite graphs are well-quasi-ordered by the minor relation. They also showed that determining whether a graph H is a minor of a graph G, parameterized by $|H|$, is *FPT* (we say that *the minor order has* FPT *order tests*) [107, 108]. These two results together show that membership in any minor-closed family of graphs can be tested in $\mathcal{O}(n^3)$ time — for a fixed family, the largest element in the obstruction set has *constant* size — and therefore provide a very powerful *FPT* classification method applicable to many graph problems. Historically, the results from the Graph Minors project provided a major impetus to the development of parameterized complexity theory (see [43]). In the remainder of the section we highlight the potential of WQO theory to provide *FPT* classifications for the parameterized problems studied by the ecology program, based on recent work by subset of the authors and Danny Hermelin.

Recall that the vertex cover (resp. feedback vertex) number of a graph is the smallest number of vertices needed to cover all edges (resp. cycles), and that the circumference of a graph is the length of its longest cycle. Graph H is a minor of G if H can be obtained from G by repeated edge deletion, edge contraction and vertex deletion. If H can be obtained by vertex/edge deletion and the contraction of edges which have at least one endpoint of degree two, then H is a *topological* minor of H. And finally, if H can be obtained using only edge contraction and vertex deletion (no edge deletion) then it is an *induced minor* of G. Of the three, only the "plain" minor relation gives a WQO in general graphs, but this changes when imposing structural restrictions on the set of graphs under consideration.

Lemma 1 ([52])**.** *For every fixed $k \in \mathbb{N}$, the following holds:*

1. *The set of all graphs of vertex cover number at most k is* WQO *by subgraphs.*
2. *The set of all graphs with feedback vertex number at most k is* WQO *by topological minors.*

3. *The set of all graphs with circumference at most k is* WQO *by induced minors.*

These well-quasi-orders guarantee the existence of finite obstruction sets for lower ideals of the set of structurally bounded graphs, under the respective relationships. To leverage these obstruction sets to establish fixed-parameter tractability results, we also need an algorithm to perform order tests under the subgraph, topological minor and induced minor relationships. In the restricted graph settings considered by the lemma, we can use Courcelle's theorem with the fact that graphs of vertex cover number, feedback vertex number, or circumference at most k, also have treewidth at most k: this leads to the existence of a $f(k+|H|)n$ algorithm for order tests between two graphs[3]. The combination of these well-quasi-orders with the FPT order tests results in the following classification tools.

Theorem 1 ([52])**.** *For every fixed $k \in \mathbb{N}$, there is a linear-time algorithm for recognizing any family of graphs with:*

1. *vertex cover number at most k that is closed under subgraphs.*
2. *feedback vertex number at most k that is closed under topological minors.*
3. *circumference at most k that is closed under induced minors.*

Let us finally give a concrete example of an ecology result that can be obtained using this machinery. Observe that the bandwidth of a graph does not increase when taking a subgraph. Hence the graphs of bandwidth at most ℓ are closed under subgraphs, resulting in the following corollary.

Corollary 1 ([52])**.** *For every fixed $k, \ell \in \mathbb{N}$ there is a linear-time algorithm which on input a graph G that has a vertex cover of size at most k, decides whether or not the bandwidth of G is at most ℓ.*

We should note that, in general, a straight-forward application of the WQO technique results in *non-uniform* algorithmics; in the preceding example, there is a *different* algorithm for every combination of k and ℓ. However, there are methods available which often make it possible to obtain a uniform algorithm at the cost of an increased running time [53, 54].

The study of general FPT classification tools, such as well-quasi-ordering, for rows of the parameter ecology matrix other than bounded treewidth, has scarcely begun. It would be interesting to investigate which rows can be systematically attacked using WQO's.

2.3.3. Protrusions to Kernelize Graph Problems with Structural Parameters

The techniques discussed in the previous sections can be used to establish the existence of FPT-algorithms, but generally do not tell you anything about

[3]Grohe et al. [68] recently showed that testing whether H is a topological minor of G can be done in $f(|H|)n^3$ time even for general graphs of unbounded treewidth.

kernel sizes. We will now see an example of a generally applicable technique which often results in *linear*-vertex kernels for graph problems in restricted graph families, under structural parameterizations. These meta-kernelization schemes revolve around the notion of *protrusions*, and are based on the idea of repeatedly replacing large yet simple parts of the graph by smaller parts which enforce the same restrictions on the existence of a (global) solution [9, 12, 60, 62].

Formally, an *r-protrusion* in a graph G is a set of vertices $X \subseteq V(G)$ such that $|N_G(X)| \leq r$ and $\text{TW}(G[X \cup N_G(X)]) \leq r$, i.e., a vertex set with at most r neighbors, whose closed neighborhood induces a graph of treewidth at most r. We will be concerned with r-protrusions where r is some constant — hence the implied bound on the treewidth of the protrusion ensures that for most problems, it is easy to analyze the behavior within the protrusion (for example using Courcelle's theorem). Given a large r-protrusion X, we would like to replace it by a smaller subgraph which has the same effect on the existence of a solution. In general, the presence of the "simple but large" protrusion X in the graph may have arbitrary effects which cannot be simulated by small replacement subgraphs; but there is a wide class of problems where pieces with a constant-size boundary can be simulated by constant-size subgraphs. This is captured by the notion of *finite integer index*, which has its roots in automata theory and reduction algorithms [17, 38]. Without going into too much details, a decision problem on graphs has finite integer index if for every constant-size separator in a graph, the number of ways in which an optimal solution on the "left" side of the separator can respond to changes in the "right" side of the separator is bounded by a function of the separator size alone. This ensures that given a sufficiently large r-protrusion, there is a smaller protrusion with the same behavior that can be substituted for it. Graph problems with finite integer index are abundant, including classic examples such as Dominating Set, Independent Set and Connected Vertex Cover. An example of a problem which does not have finite integer index is Longest Path.

Before explaining more of the technique, let us describe what kinds of parameterizations we are attacking. In the literature, these meta-kernelization techniques are applied to the natural parameterizations of *bidimensional* problems. But since the focus of this survey is on the ecology of parameters, we explain the ideas for structural parameterizations[4]. So pick your favorite constant t, and let the (vertex-)deletion distance of graph G to treewidth t be defined as the size of a smallest set $X \subseteq V(G)$ such that $\text{TW}(G - X) \leq t$. Note that for treewidth 0 and 1, this coincides with the vertex cover number and feedback vertex number. The meta-kernelization techniques can be applied to problems parameterized by the deletion-distance to constant treewidth, regardless of the constant.

[4]This is effectively what is happening under the hood of the existing approaches, where the bidimensionality condition and a restriction on the class of input graphs together ensure that YES-instances for a value k of the natural parameter, have vertex-deletion distance $\mathcal{O}(k)$ to a graph of constant treewidth.

So how does a replacement strategy yield polynomial kernels for such structural parameterizations? For the protrusion-replacement strategy to be effective as our sole means of obtaining small kernels, we need to ensure that the graph has become small with respect to our structural parameter value once all remaining protrusions are small. Unfortunately, this is not the case in general: for all constants t and r there are graphs with deletion distance k to treewidth t in which all r-protrusions have constant size, yet whose total size is unbounded in k; a concrete example is the complete bipartite graph $K_{k,n}$ with $n, k > r$, in which any r-protrusion has size at most r, but which is arbitrarily large compared to its vertex-deletion distance to treewidth zero (since deleting the k vertices on the left side leaves an independent set). That is why the framework is applied to graphs from restricted graph classes. While originally used in the setting of planar and bounded-genus graphs, it has since been generalized to H-minor-free graph classes for arbitrary H [63]. The restriction to such sparse graph families is sufficient to make the approach work: it can be proven that for every graph H, there is a constant r' such that if G is H-minor-free, has deletion distance k to constant treewidth, and its largest r'-protrusion has constant size, then $|V(G)| \in \mathcal{O}(k)$ (for fixed H and r').

Combining the protrusion reduction rule with the size bound on H-minor-free graphs in which all r'-protrusions have constant size, yields linear-vertex kernels for all problems with finite integer index parameterized by the deletion distance to constant treewidth (regardless of what the constant is), when the inputs graphs are H-minor-free for some H. Thus the protrusion replacement technique yields linear-vertex kernels for a wide range of problems under structural parameterizations, albeit only on restricted graph families. Let us remark at this point that there are good reasons why the framework fails to yield polynomial kernels on general graphs when parameterized by the deletion distance to constant treewidth: for several basic graph problems such as Vertex Cover [34], Feedback Vertex Set [79] and Hamiltonian Cycle [14] this has been proven to be impossible, subject to $NP \not\subseteq coNP/\text{poly}$, even when parameterized by the size of a given deletion set to treewidth two.

2.3.4. Other techniques

Finally, let us briefly mention two other methods which have been used successfully in the past to attack problems under structural parameterizations. A famous theorem by H. W. Lenstra jr. shows that testing the feasibility of an Integer Linear Program (ILP) in n variables is FPT parameterized by n. Successive improvements to his theorem [64, 81] have been used to establish the fixed-parameter tractability of voting problems [50] and various graph layout problems parameterized by vertex cover number [56]. We refer to Niedermeier's book for more examples of the applicability of the technique, such as Closest String [99, Chapter 11.2].

Many graph problems admit linear-vertex kernels when parameterized by the maximum number of leaves in a spanning tree for the input graph — or the sum of these numbers over all connected components, in case the input is disconnected — which can be explained by a theorem due to Kleitman and

West [84]. Their elegant result states that a graph on n vertices, with minimum degree at least three, has max leaf number at least $\lceil \frac{n}{4} + 2 \rceil$. A corollary to this theorem essentially shows that the only reason a large graph can have a small max leaf number, is because there are long paths of degree-2 vertices. For all graph problems in which such long degree-2 paths can be shrunk to a constant size (in particular, all problems of Table 2 except BANDWIDTH), this leads to a linear-vertex kernel [14, 49, 55].

3. A Refined View of Ecology: Studying a Hierarchy of Parameters

Take a breath and reflect on what has been discussed so far. We advocated the investigation of how various structural notions interact with our computational objectives — both due to theoretical interest (it tells us how various aspects of a problem contribute to its difficulty) and due to practical relevance (as it may result in one of the two positive deliverables for parameters which are small in real-world instances). We organized these studies in a table, dealt with the proper formalization of our questions and looked at systematic ways to attack rows of this table. Now that we have motivated the ecology program and given some examples of what it entails, it is time to take the next step and refine our view of parameter ecology.

3.1. Relationships Between Parameters

Is the table perspective the best way of organizing our assault on intractability? One might wonder whether all the entries in this infinite table of problems versus parameters are really interesting. Are we just creating more work for ourselves, studying toy problems? Indeed, not all the entries in our table hold the same appeal. For example, many of the *FPT* classifications for the BANDWIDTH row of Table 2 were based on the fact that the bandwidth of a graph bounds its treewidth, together with *FPT* algorithms for bounded treewidth: these entries told us nothing new. What the table fails to take into account are the *relationships* between various parameters, and it is about time we remedy the situation. Fig. 1 therefore organizes structural parameters of graphs into a hierarchy, based on the partial ordering on them. The parameters in the hierarchy correspond to the optimum values of classic optimization problems on graphs, structural measures, and to the (vertex-)deletion distance to well-known graph classes (see [8, 18] for definitions). For example, the parameter *Odd Cycle Transversal* is the size of the smallest vertex set intersecting all odd-length cycles, and the *Distance to Linear Forest* is the minimum number of vertices whose removal results in a forest of maximum degree at most two. The figure gives relationships between these parameters: we see that graphs of vertex cover number k or max leaf number k both have treewidth $\mathcal{O}(k)$, while the vertex cover number of a graph can be arbitrarily large compared to its max leaf number, and vice versa.

The aforementioned *FPT* classification for BANDWIDTH parameterized by treewidth is an example of the propagation of results through the hierarchy: if

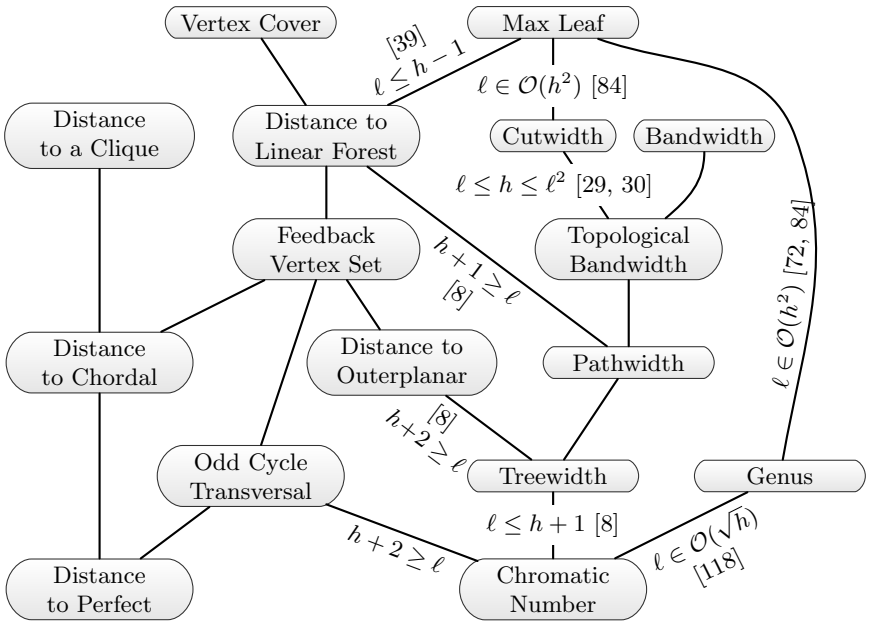

Figure 1: A hierarchy of parameters, with larger parameters drawn higher. An unlabeled line between two parameters means that the parameter drawn lowest is never larger than the one drawn highest. These unlabeled relationships (cf. [109]) follow from inclusions between graph classes [18] or bounds which can be found in Bodlaender's survey on treewidth [8]. When a line between parameters is labeled by a bound, the lower-drawn parameter is represented by ℓ and the higher-drawn parameter by h.

a problem is *FPT* parameterized by $\pi(G)$ and parameter $\pi'(G)$ is larger in the sense that there is a function f such that $f(\pi'(G)) \geq \pi(G)$ for all graphs G, then an FPT-algorithm parameterized by $\pi(G)$ also yields fixed-parameter tractability[5] under $\pi'(G)$. Similarly, a $W[1]$-hardness proof for some parameterization implies that all smaller parameterizations are also $W[1]$-hard. When the relationships between the parameters are polynomial (which all bounds in Fig. 1 are) then the (non-)existence of polynomial kernelizations propagates through the hierarchy as well: positive results transfer upwards to larger parameters, and negative results carry over to smaller parameters.

Using the propagation of results by the given relationships, the hierarchy *guides* us in our attack on intractability, pointing us to interesting questions and telling us over which flank a problem may be attacked. For example, as we know that Bandwidth is NP-complete on trees [67, GT40] which have trivial feedback vertex sets of size zero, we know that no parameterization below *Feedback Vertex Set* can be *FPT* unless $P = NP$; hence we should look for

[5]In light of our discussion in Section 2.2 we should of course demand that the two parameterizations are similarly formalized, which we implicitly assume at this point.

FPT results for parameters which are larger than, or incomparable to, the feedback vertex number. When also taking the $W[t]$-hardness of the natural parameterization into account, the hierarchy shows that the remaining viable regions for tractable parameterizations are the vertex-deletion distance to a single clique, the distance to a linear forest, or the vertex cover/max leaf number. And indeed, early results in the parameter ecology program have shown fixed-parameter tractability for the latter two [55, 56]. The hierarchy now points us to an interesting open problem: since fixed-parameter tractability parameterized by the distance to a linear forest would imply the tractability by vertex cover and max leaf number, the former parameterization has the potential to unify and generalize two fixed-parameter tractable cases of Bandwidth.

Open Problem 6. Is Bandwidth in FPT[6] parameterized by the deletion distance to a linear forest?

The idea of considering "larger" parameterizations when encountering intractability has been applied in a few cases; especially regarding the parameter treewidth. Thanks to recent work we now know that besides Bandwidth, there are various other problems which are $W[1]$-hard for the parameter treewidth. Among them are List Coloring, Pre-Coloring Extension and Equitable Coloring [51]; General Factor [114]; and Minimum Maximum Outdegree [113]. Not all these problems behave the same using larger parameterizations like vertex cover number: although Pre-coloring Extension and Equitable Coloring become fixed-parameter tractable, List Coloring remains $W[1]$-hard for graphs of bounded vertex cover number [58].

These examples illustrate that the questions which the parameter hierarchy implores us to ask are not about toy problems, but are well motivated due to the inherent difficulty of the problems: $W[1]$-hardness of the parameterization by treewidth *forces us* to resort to larger parameters when looking for fixed-parameter tractability. The unfortunate fact of nature that the problem is hard for smaller parameterizations leaves us little other choice (besides taking an *additional* parameter, will be discussed in Section 5.1).

3.2. Establishing a New Race: Pushing Towards the Boundaries of Tractability

Although *FPT* classifications by vertex cover number give us some grip on problems such as Bandwidth which are hard parameterized by treewidth, they *should in no way be the end* of our investigations. As the hierarchy shows, there is a range of parameters between vertex cover number and treewidth and it is *always* meaningful to find out whether positive news (in the form of FPT-algorithms or polynomial kernels) can be transferred to smaller parameters — *even* if this (initially) requires a worse dependency on the parameter in the run-time or kernel size.

[6] As the deletion distance to a linear forest can be computed in FPT-time, using Courcelle's theorem and the implied bound on the treewidth, the choice between the three types of formalizations mentioned in Section 2.2 has no effect on the answer to this question.

Consider two graph parameters $\pi'(G)$ and $\pi(G)$ such that $\pi'(G)$ is *structurally smaller* than $\pi(G)$ in the sense that there is a function f such that $\pi'(G) \leq f(\pi(G))$ for all graphs, whereas there is no g such that $\pi(G) \leq g(\pi'(G))$ on all graphs[7]. For all such pairs of parameters, there are (under mild technical conditions which are satisfied for NP-hard problems) inputs on which *any* FPT-algorithm or polynomial kernel with respect to π' is better than algorithms with respect to π, regardless of the growth rates of the functions involved. The structural difference between the parameters implies that there are inputs where a *bigger* function of the *smaller* parameter yields a better overall run-time than the smaller function on the larger parameter. Similarly, there will be inputs where the kernel bound guaranteed by a bigger (yet polynomial) function of a smaller parameter beats the bound which is guaranteed for the larger parameterization. These observations motivate the search for positive FPT-deliverables for *increasingly smaller parameters*. Another justification for the search of good news for smaller parameterizations, even at the cost of bigger functions $f(k)$ or kernel sizes, is the observed trend that once good news has been established, successive improvements are usually found that decrease the run-time and kernel size[8].

These ideas call for a new type of *race* in parameterized complexity analysis: the race for the *best* (i.e., smallest) parameter with respect to which the problem is still *FPT*, or still admits a polynomial kernel. Only if we know where the boundary lies between *FPT* and $W[1]$-hardness in the parameter landscape, do we fully understand what constitutes the difficulty of the problem. No parameterized analysis of a problem should therefore be complete without asking: can we extend our positive news to a smaller parameter? There have been interesting improvements in this regard for the Vertex Cover problem. The existence of a kernel with $2k$ vertices for the naturally parameterized k-Vertex Cover problem has been known for a decade [25]. But it was recently shown that Vertex Cover parameterized by the size ℓ of a given feedback vertex set admits a kernel with $\mathcal{O}(\ell^3)$ vertices [77]. For a practical example of the relevance of this smaller parameter, consider a path on n vertices: it has vertex cover number $k = \lfloor n/2 \rfloor$ but feedback vertex number zero. The kernelization with respect to the natural parameterization promises to shrink an instance (P_n, k) to an equivalent instance on at most $2k \approx n$ vertices; the instance is not guaranteed to shrink at all. But as the parameter feedback vertex number is zero, the kernelization with respect to that parameter promises that the instance will

[7]This holds for all relationships exhibited in the hierarchy of Fig. 1, with the exception of the pair Cutwidth and Topological Bandwidth since one is *sandwiched* by the other: $\text{tbw}(G) \leq \text{cw}(G) \leq \text{tbw}(G)^2$. This phenomenon has interesting applications in parameterized approximation. For concreteness of the discussion at hand, one could think of $\pi'(G)$ as treewidth, and $\pi(G)$ as the vertex cover number.

[8]There are, however, some intriguing cases where the function $f(k)$ for a smaller parameter provably *cannot* be decreased to match the dependency function of a larger parameter: compare the non-elementary lower-bound by Frick and Grohe [65] for MSOL model-checking parameterized by treewidth to the (doubly)exponential upper bounds parameterized by vertex cover and max leaf number, as given by Lampis [88].

be shrunk to *constant* size! It now seems as if the polynomial kernelization by feedback vertex number was just the tip of the iceberg; further milestones in the race for the best Vertex Cover parameterizations form the topic of Section 4.

The existence of a polynomial kernel for a "better-than-natural" parameterization of Vertex Cover raises the question whether the same exists for Feedback Vertex Set, which is one of the other big success-stories of kernelization.

Open Problem 7. Is there a structural parameter smaller than feedback vertex number with respect to which Feedback Vertex Set admits a polynomial kernel?

Since the feedback vertex number measures the vertex-deletion distance to treewidth one, a natural choice of smaller parameter would be the distance to treewidth two. Alas, we shall have to look elsewhere for a positive solution to the problem: assuming $NP \not\subseteq coNP/\text{poly}$ the Feedback Vertex Set problem does not admit a polynomial kernel parameterized by deletion distance to an outerplanar graph (which is a larger parameter than distance to treewidth two), which follows by a simple adaptation of the kernel lower bound for Odd Cycle Transversal [80].

Of course, the quest to extend positive news to smaller parameterizations is only one side of the story: we should consider simultaneously whether negative results can be lifted to larger parameters. When the upwards and downwards motions meet, we have succeeded in uncovering the boundary of tractability for the problem under consideration. Notable examples of results in parameter ecology include work on the Graph Coloring problem [20, 78, 90], on problems parameterized by the "number of gaps in an interval representation" [61], refined parameterizations for Two Layer Planarization [115], and on parameterizations by Twin Cover [66], a parameter sandwiched between treewidth and vertex cover number. The next section gives a detailed report on advances for the Vertex Cover problem.

4. Case Study: Milestones in the Analysis of Vertex Cover

In this section we survey recent developments in parameter ecology for the Vertex Cover problem. True to our own advice of Section 3.1, we shall start by considering the relationships between the parameters which are relevant for this problem; we will encounter some parameters which were not present in Fig. 1. After getting a feel for these parameters we discuss the most recent results with respect to them.

4.1. Interplay Between Vertex Cover *Parameterizations*

To truly appreciate the advancements which have been made for Vertex Cover we need to spend some time on the relationships between the various parameters. Although this material might seem a bit dry at first glance it is of vital importance, which can be inferred from the fact that — as we shall

see later on — merely this knowledge of the interactions between parameters is sufficient to answer an open problem posed by Gutin et al. [71].

Some interesting parameterizations of Vertex Cover are based on the idea of parameterizing *above or below tight bounds*. For an undirected graph G, let $\mu(G)$ denote the cardinality of a maximum matching and let the vertex cover number of G be $\text{VC}(G)$. If M is a matching, then a vertex cover contains at least one endpoint of every edge in M. Hence all graphs satisfy $\text{VC}(G) \geq \mu(G)$, and it is easy to find graphs where equality holds. Rather than using the requested size of the vertex cover as the parameter, one may therefore attempt a parameterization by the excess of the size of a vertex cover over the matching number: one could ask whether G has a vertex cover of size $\mu(G) + k$, parameterized by k, and the resulting values of k are smaller than the natural parameter. To place the parameterization by this "above guarantee" value k in our hierarchy, we cannot rely on some input value k: we consider the corresponding structural measure $\text{VC}(G) - \mu(G)$, which will give us the right feeling for the strength of the parameterization.

We can also parameterize *below* a tight bound. Let $\gamma'(G)$ be the minimum cardinality of a maximal matching[9]. For any maximal matching $M \subseteq E(G)$, the $2|M|$ endpoints of the matching edges form a vertex cover — else the matching would not be maximal — hence $\text{VC}(G) \leq 2\gamma'(G) \leq 2|M|$. Given a maximal matching M and graph G, one can ask whether G has a vertex cover of size $2|M| - k$, parameterizing below the guaranteed upper bound. To relate this parameterization to others, we again turn to a structural measure of graphs. In this case the relevant measure is $2\gamma'(G) - \text{VC}(G)$, and since $\gamma'(G) \leq \mu(G) \leq \text{VC}(G)$ it satisfies $2\gamma'(G) - \text{VC}(G) \leq \text{VC}(G)$: it is a smaller measure than the vertex cover number.

Proposition 1 illustrated the relevance of parameterizations by the vertex-deletion distance to graph classes in which Vertex Cover is polynomial-time solvable. It turns out [95] that an interesting role is played by the König (-Egerváry) graphs — these are the graphs that satisfy $\text{VC}(G) = \mu(G)$. König's famous theorem states that all bipartite graphs satisfy this condition, which shows that the deletion distance to a König graph cannot exceed the distance to a bipartite graph. In other words, the size of an odd cycle transversal bounds the distance to a König graph. This parameter also has an interesting connection to the parameterization above lower bound that we encountered earlier.

Lemma 2 ([95, Lemma 12])**.** *Let G be a graph with deletion distance k to a König graph. Then $\text{VC}(G) - \mu(G) \leq k \leq 2(\text{VC}(G) - \mu(G))$.*

The next parameterization we consider — one of the current best in the parameter hierarchy for which positive results are known — also takes the form of an "above lower bound" question. It revolves around the linear-programming relaxation of Vertex Cover: the linear program with non-negative real-valued variables w_v for $v \in V(G)$ that seeks to minimize $\sum_{v \in V(G)} w(v)$, with a con-

[9] This value coincides with the edge domination number, which explains the notation.

straint $w_u + w_v \geq 1$ for every edge $\{u, v\} \in E(G)$. Feasible solutions to this LP are also called fractional vertex covers, and the minimum value of a valid solution is denoted by $\text{LP}(G)$. By the nature of relaxation, all graphs satisfy $\text{LP}(G) \leq \text{VC}(G)$; the difference can be arbitrary. As before, we may ask whether G has a vertex cover of size $\text{LP}(G) + k$, parameterized by k. This boils down to the study of the related structural measure $\text{VC}(G) - \text{LP}(G)$. It is related to several parameters we saw before, as was proven by Kratsch and Wahlström in their recent work [86]. To give an accurate picture of the relationships between parameters, we show that $\text{VC}(G) - \mu(G) \leq 2(\text{VC}(G) - \text{LP}(G))$, improving on their bound. The following classic result due to Nemhauser and Trotter is the key to the proof.

Theorem 2 ([4, 83, 96])**.** *For every graph G there is an optimal* half-integral *solution to the LP-relaxation of* Vertex Cover *on G in which all vertices are assigned values from the set $\{0, 1/2, 1\}$. Furthermore, if $S_0, S_{1/2}, S_1$ are the sets of vertices with value $0, 1/2$ and 1 in an optimal fractional vertex cover, then there is a minimum (integral) vertex cover X of G such that $S_1 \subseteq X \subseteq S_1 \cup S_{1/2}$, i.e., a vertex cover containing all vertices of S_1 and none of S_0.*

Lemma 3. *Every graph G satisfies $\text{VC}(G) - \mu(G) \leq 2(\text{VC}(G) - \text{LP}(G))$.*

Proof. Fix some optimal half-integral solution I to the LP-relaxation of Vertex Cover on G, and let $S_0, S_{1/2}, S_1$ be the sets of vertices which obtain the values indicated by their respective subscripts. Let $X \subseteq V(G)$ be a minimum vertex cover of G such that $S_1 \subseteq X \subseteq S_{1/2} \cup S_1$ whose existence is guaranteed by the Nemhauser-Trotter theorem. Let $X_{1/2} := X \cap S_{1/2}$, which implies that $X = X_{1/2} \cup S_1$. We show how a large matching in G can be constructed in two parts.

Let us first show that S_1 can be matched into S_0, i.e., that the bipartite graph G' with partite sets S_1 and S_0 and edge set $\{\{u, v\} \in E(G) \mid u \in S_0 \wedge v \in S_1\}$ has a matching saturating all of S_1. Assume for a contradiction that there is no such matching in G'. By Hall's theorem [110, Theorem 16.7] this guarantees the existence of a set $S_1' \subseteq S_1$ such that $|N_{G'}(S_1')| < |S_1'|$. Now consider the fractional vertex cover of G which is obtained from I by decreasing the value of all vertices in S_1' from 1 to $1/2$, and increasing the values for $N_{G'}(S_1')$ from 0 to $1/2$; it is easy to verify that this strictly decreases the total weight of the solution. For every vertex in S_1' whose weight is lowered, the weight of all its neighbors is either already at least half, or explicitly increased to $1/2$; hence this altered solution satisfies all constraints, contradicting the optimality assumption for I. Therefore it is possible to match S_1 into S_0, which gives us the first part of a matching.

For the second part of the matching we will construct, define $Y_{1/2} := S_{1/2} \setminus X_{1/2}$. We show that $Y_{1/2}$ can be matched into $X_{1/2}$, i.e., that the bipartite graph G'' with partite sets $Y_{1/2}$ and $X_{1/2}$ has a matching saturating $Y_{1/2}$. For a contradiction similar as before, assume that no such matching exists. Let $Y_{1/2}' \subseteq Y_{1/2}$ be a Hall set such that $|N_{G''}(Y_{1/2}')| < |Y_{1/2}'|$. The fractional vertex cover obtained from I by increasing the value of all vertices in $N_{G''}(Y_{1/2}')$ from $1/2$ to 1, and decreasing the values assigned to $Y_{1/2}'$ from $1/2$ to 0, has smaller

weight than I. As the vertices in $Y_{1/2}$ form an independent set in G (they are a subset of a complement of a vertex cover), they have all their G-neighbors in the set $N_{G''}(Y'_{1/2}) \cup S_1$, which implies that all vertices for which the weight is decreased to 0 end up with all their neighbors having weight at least 1. So the result is a valid fractional vertex cover, again contradicting optimality of I: we conclude that $Y_{1/2}$ can be matched into $X_{1/2}$.

These two pieces of information allow us to lower-bound the size of a maximum matching in G:

$$\begin{aligned}
\mu(G) &\geq |S_1| + |Y_{1/2}| && \text{Match } S_1 \text{ to } S_0,\ Y_{1/2} \text{ to } X_{1/2}.\\
&= |S_1| + |S_{1/2}| - |X_{1/2}| && \text{Since } S_{1/2} = X_{1/2} \cup Y_{1/2}.\\
&= 2(|S_1| + |S_{1/2}|/2) - (|S_1| + |X_{1/2}|)\\
&= 2\,\textsc{lp}(G) - \textsc{vc}(G).
\end{aligned}$$

The last steps are justified by noting that the fractional solution has cost exactly $|S_1| + |S_{1/2}|/2$ by definition of the sets S_i, and that the minimum vertex cover X has cardinality $|S_1| + |X_{1/2}|$ since it avoids S_0. The inequality implies the lemma by simple formula manipulation. □

The bound given by Lemma 3 is best-possible, which is witnessed by graphs which are disjoint unions of triangles. In the other direction it is easy to see that $\textsc{vc}(G) - \textsc{lp}(G) \leq \textsc{vc}(G) - \mu(G)$, as the value of the LP-relaxation must be at least the size of a maximum matching: for every edge in the matching, the two endpoints of the edge together have value at least one.

Before we start the discussion of recent results, there is one more "above guarantee" parameterization to consider. Let $\Delta(G)$ be the maximum degree of a vertex in G. For readability, we will write Δ instead of $\Delta(G)$, and m instead of $|E(G)|$ in the remainder. Since a single vertex can cover at most Δ edges, any vertex cover has size at least m/Δ, and there are graphs where this is sufficient. So we may ask whether G has a vertex cover of size $m/\Delta + k$ parameterized by k, giving rise to the structural measure $\textsc{vc}(G) - m/\Delta$. It is not difficult to show how this measure relates to the excess of an integral vertex cover over a fractional vertex cover.

Proposition 2. *Every graph G satisfies* $\textsc{vc}(G) - \textsc{lp}(G) \leq \textsc{vc}(G) - m/\Delta$.

Proof. Fix some fractional vertex cover, and observe that $\Delta \cdot \textsc{lp}(G) = \Delta \cdot \sum_{v \in V(G)} w_v \geq \sum_{\{u,v\} \in E(G)} w_u + w_v \geq \sum_{\{u,v\} \in E(G)} 1 = m$. This implies the claim by simple formula manipulation. □

We have now established the relationships needed to annotate a proper hierarchy containing the parameters relevant to this section: it is given in Fig. 2. Let us interpret recent results in the light of these interactions.

4.2. Parameterized Complexity of Vertex Cover

To make sense of the multitude of results for this *drosophila* of parameterized algorithmics, we describe the advances one parameter at a time. It has

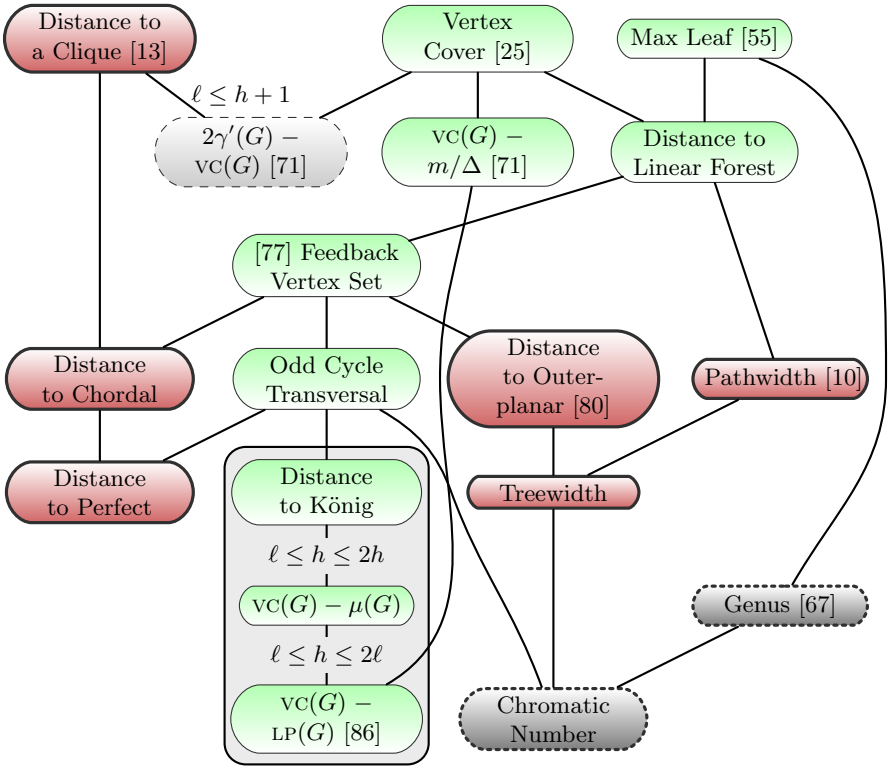

Figure 2: Complexity overview for various parameterizations of Vertex Cover, assuming suitable formalizations. For deletion distance parameters we assume that a deletion set is given along with the input. The shading indicates that a parameterization is either NP-complete for constant values of the parameter, $W[1]$-hard but contained in XP, FPT but (conditionally) lacking a polynomial kernel, or FPT with a polynomial (randomized) kernel. Relationships between parameters present in Fig. 1 are not repeated. The three parameters grouped at the bottom are equivalent up to constant factors. The proofs for the bounds not mentioned explicitly in the main text are easy exercises.

been known for decades that the natural parameterization k-Vertex Cover is FPT [19], the current-best run-time being $\mathcal{O}(1.2738^k + kn)$ which was obtained after a series of improvements [25, 26, 46, 101, 102]. k-Vertex Cover admits a kernel with $2k$ vertices, which can be obtained using the Nemhauser-Trotter theorem [25, 96], or by crown reductions [1, 27, 28]. The existence of smaller kernels has been subject of repeated study. Chen et al. [24, Corollary 3.13] showed that for every $\varepsilon > 0$, no polynomial-time algorithm can reduce instances (G, k) of k-Vertex Cover to equivalent instances (G', k') on at most $(2-\varepsilon)k$ vertices such that G' is a subgraph of G and $k' \leq k$, unless $P = NP$. A breakthrough result by Dell and van Melkebeek [40] showed that the number of edges in the kernel (which is quadratic in the number of vertices) cannot be significantly improved either: unless $NP \subseteq coNP/\text{poly}$ (which implies a collapse of the polynomial hierarchy [119]), there is no kernel for k-Vertex Cover which reduces

an instance to an equivalent one of bitsize $\mathcal{O}(k^{2-\varepsilon})$ for any $\varepsilon > 0$. It is possible to shave a little bit off the kernel size, though: Soleimanfallah and Yeo [112] showed that for every constant c there exists a kernel with $2k - c$ vertices. This is mostly of theoretical interest however, since the running time of the kernelization algorithm is exponential in c. Their approach was extended by Lampis [87], who obtained a kernel with $2k - c \log k$ vertices for any c by exploiting a connection to Almost 2-SAT.

This connection was first uncovered by Mishra et al. [93–95] when studying the influence of the deletion distance to König graphs on the complexity of Vertex Cover: they established the FPT-equivalence of Vertex Cover Above $\mu(G)$ and Almost 2-SAT. Razgon and O'Sullivan later found an FPT-algorithm for the latter problem, thereby placing Vertex Cover Above $\mu(G)$ in *FPT*. Successive improvements to the running time quickly followed each other [37, 105], the current record-holder being an algorithm by Cygan et al. which runs in $\mathcal{O}(4^k n^c)$ time. Through their seminal use of matroid techniques for kernelization, Kratsch and Wahlström obtained a *randomized* polynomial kernel for Vertex Cover Above $\mu(G)$ [86]. By relating the measure $\text{vc}(G) - \text{lp}(G)$ to $\text{vc}(G) - \mu(G)$, they showed the same good news holds for Vertex Cover Above $\text{lp}(G)$ which is currently one of the strongest parameterizations known to be tractable. It came as a surprise that this parameterization admits a polynomial kernel: we often find that the boundary for polynomial kernelizability lies higher in the hierarchy than the threshold for solvability in FPT-time.

Such a separation is witnessed for example when considering more traditional measures of graph structure such as treewidth. As Vertex Cover can be formulated in Monadic Second Order Logic (even without using quantifications over edge sets), it is *FPT* parameterized by treewidth [16] and even smaller parameters such as cliquewidth [32]. These *width measures* (parameters which do not increase when making multiple disjoint copies of a graph) generally do not yield parameterizations admitting polynomial kernels, as was indeed proven to be the case for Vertex Cover parameterized by treewidth [10]; the proof assumes $NP \not\subseteq coNP/\text{poly}$, and also applies to pathwidth.

Another branch of parameterizations can be interpreted as *parameterizing away from triviality* (cf. [20, 99, 100]). If the input graph is "close" to a graph class $\mathcal{F}$ in which Vertex Cover is polynomial-time solvable, then one expects to be able to exploit this connection to solve the problem more efficiently. Formalizing closeness as the vertex-deletion distance to $\mathcal{F}$, the proof technique of Proposition 1 indeed shows that if $\mathcal{F}$ is hereditary and a deletion set to $\mathcal{F}$ of size k is given, then the problem can be solved in $\mathcal{O}(2^k n^c)$ time. As the feedback vertex number measures the deletion distance to a forest in which the problem is easily solvable, the kernelization for the corresponding parameterization [77] which was already mentioned in Section 3.2 shows how this distance to triviality can be exploited.

There is a remarkable gap between polynomial kernelizability and fixed-parameter tractability for such parameterizations away from triviality. If $\mathcal{F}$ contains all cliques, then Vertex Cover parameterized by distance from $\mathcal{F}$ does not admit a polynomial kernel unless $NP \subseteq coNP/\text{poly}$ [13], while we saw

before that even the distance to the much richer class of perfect graphs yields FPT-algorithms. The difficulty of data reduction for these parameters cannot just be explained by the density of the graphs in $\mathcal{F}$: considering sparser and simple classes $\mathcal{F}$ such as the graphs of treewidth two [34], or even outerplanar graphs[10], the corresponding parameterizations admit no polynomial kernels.

With regards to parameterizations above and below tight bounds, Gutin et al. [71] proved that when given a maximal matching M in graph G and an integer k, it is $W[1]$-hard to decide whether G has a vertex cover of size $2|M|-k$ parameterized by k. In the same paper they posed the existence of an FPT-algorithm for the following parameterization as an open problem:

> Vertex Cover Above m/Δ
> **Instance:** Graph G with m edges and maximum degree Δ, and an integer k.
> **Parameter:** k.
> **Question:** Does G have a vertex cover of size at most $m/\Delta + k$?

We will now illustrate the importance of the bounds obtained in Section 4.1 by making good on the promise of showing how they answer an open problem.

Theorem 3. Vertex Cover Above m/Δ *is* FPT *and admits a randomized polynomial kernel.*

Proof. Let (G, k) be an instance of Vertex Cover Above m/Δ. Compute a maximum matching in G, and let $k' := m/\Delta + k - \mu(G)$. If $k' < 0$ then the instance asks for a vertex cover of size less than that of a matching; hence we may safely output no. We will show that if the input consists of a yes-instance, then $k' \leq 2k$. To see this, observe that if (G, k) is a yes-instance then $m/\Delta + k \geq \text{vc}(G)$, so:

$$k \geq \text{vc}(G) - m/\Delta.$$

The combination of Lemma 3 and Proposition 2 shows that $\text{vc}(G) - \mu(G) \leq 2(\text{vc}(G) - m/\Delta)$. By elementary manipulation this implies:

$$\text{vc}(G) - m/\Delta \geq m/\Delta - \mu(G).$$

Combining the two inequalities shows that $k \geq m/\Delta - \mu(G)$. Adding k to both sides gives:

$$2k \geq m/\Delta + k - \mu(G) = k',$$

which indeed proves that yes-instances satisfy $k' \leq 2k$. Hence after computing the value of k', we may safely output no if it exceeds $2k$. In the remaining cases we make a copy G' of G, and interpret (G', k') as an instance of Vertex Cover Above $\mu(G)$. It is trivially equivalent to the input instance, and as the problem we reduce to is *FPT* [37, 95, 105, 106], and admits a randomized polynomial kernel [86] the theorem follows. □

[10] This follows from an unpublished construction similar to the kernel lower bound for Odd Cycle Transversal parameterized by the distance to outerplanarity [79, 80].

This concludes our discussion of the milestones of Vertex Cover research. By propagating the cited results through the hierarchy as described in Section 3.1, we arrive at the complexity picture of Fig. 2. We finish the section with some open problems related to this material.

Open Problem 8. Find a deterministic, polynomial kernel for Vertex Cover Above $\mu(G)$.

Open Problem 9. Find a parameterization which is structurally smaller than $\text{vc}(G) - \text{lp}(G)$ (in the sense of Section 3.2) with respect to which Vertex Cover is *FPT*.

After the discussion of these technical matters, we return to programmatic contemplations.

5. Towards Fully Multivariate Algorithmics

5.1. Model Enrichment vs. Deconstruction of Intractability

For practitioners who have to solve *NP*-hard problems on a daily basis, it is good to know that there are parameterized approaches to tractability which may help them to cope with the computational difficulties they are facing. Many hard problems described in the appendix of Garey and Johnson's famous monograph [67] have been successfully attacked by the FPT paradigm, resulting in clever techniques to solve them more efficiently for bounded parameter values. But what would help practitioners even more than FPT-algorithms and kernels for the *cleanly abstracted* mathematical problems around which the theory of *NP*-completeness has developed, would be to show them how to solve the problems they are actually dealing with! The problems described by Garey and Johnson are not the problems encountered in real life; rather they are simplified and idealized into slick, simple definitions. For the study of intractability, this was perfectly suitable: if the idealized versions of real-life problems are *NP*-complete, then surely the real, more complicated, problems are no easier. But when we are in the business of identifying fixed-parameter tractability, we are in the fortunate situation of having *positive* news to report. Suddenly, there is no valid justification anymore for studying simplified, dumbed-down problem settings: if an abstract version of the problem is *FPT*, we should try to establish the same result for more general formulations of the problem! After gaining an understanding of the difficulty of the problem by establishing its boundaries of tractability in the parameter hierarchy, we should, in the interests of greater realism and a better fit to applications, *explore more richly structured variations* of the problem and try to extend our positive results to these generalized settings. For example, a common type of model enrichment is to include weights into a problem formulation. Such weights allow real-world problems to be modeled more accurately, for example by using them to express differing costs among the combinatorial objects. The study of weighted problems and generalized models has permeated the approximation community, which is another branch

of theoretical computer science where *positive* results are obtained for difficult problems (c.f. [33]). In sharp contrast, the investigation of the parameterized complexity of enriched models is still in its infancy. Notable examples of the parameterized analysis of richer problem settings include the results of Marx et al. [92] applicable to various forms of constrained separation problems, work on Group Feedback Vertex Set [36, 86] which generalizes the well-studied problems of (Subset) Feedback Vertex Set, Multiway Cut and Odd Cycle Transversal, and work on weighted problems [6, 23, 27, 57, 76].

Unfortunately, we may quickly find ourselves back in the land of intractability when generalizing our problem definitions, even for parameterizations which were *FPT* in the original setting. Recall for example the Graph Coloring problem from Section 3.1, which is *FPT* parameterized by treewidth while its generalization List Coloring is already $W[1]$-hard parameterized by vertex cover number [58]. One could resort to parameters which are even larger than the vertex cover number, but it seems implausible that these would be small on typical instances. The hardness result shows that the difficulty of List Coloring is not just caused by a rich structure of the graph, as the problem remains hard when imposing severe bounds on this structure: the restricted choice of colors for the vertices, as given by the list function, has introduced a new source of difficulty for the problem which was not present in the simpler Graph Coloring model. Instead of moving to larger parameters, we could try to *deconstruct the intractability* of the problem by taking an *additional* parameter to bound the difficulties introduced by the list function. Extra parameters which lead to fixed-parameter tractability can typically be extracted from the hardness proof by asking: "Why are the instances created in the proof *unreasonable*?". An examination of the negative result(s) [51] shows that the reduction from Multicolored Clique produces instances of List Coloring where "most" vertices have lists of allowable colors of size 2, whereas the total universe of allowed colors is large. This is clearly unreasonable, and prompts us to consider the maximum number of colors which are "forbidden" for a vertex as an additional parameter.

> List Coloring with Bounded Treewidth and Deficiency
> **Instance:** A graph G, a width-k tree decomposition of G, an integer q, and a list function $L\colon V(G) \to \{1, \ldots, q\}$.
> **Parameter:** The value $k + d$, where $d := \max_{v \in V(G)} q - |L(v)|$ is the maximum *deficiency* of a list.
> **Question:** Is there a coloring function $f\colon V(G) \to \{1, \ldots, q\}$ such that $f(v) \in L(v)$ for all vertices v, and adjacent vertices receive different colors?

To illustrate the point, we offer the following simple *FPT* classification.

Theorem 4. List Coloring with Bounded Treewidth and Deficiency *is* FPT.

Proof. We show that for no-instances the total number of colors q is bounded by $d + k$. Assume therefore that $q \geq d + k + 1$. By definition of the deficiency,

this implies that every vertex has at least $k+1$ colors on its list. As G has treewidth at most k, there is an elimination order of G in which each vertex has degree at most k upon elimination [8, Section 7]. So suppose $v_1, v_2, \ldots, v_n$ is a permutation of the vertices such that each vertex has at most k neighbors which come later in the ordering. Now process this ordering from back to front, greedily assigning each vertex a color from its list while removing that color from the lists of its neighbors. Each assignment decreases the size of the lists of its neighbors by at most one. When we process vertex v_i, we have only assigned colors to vertices with indices $j > i$. Hence the property of the elimination order ensures that at most k colors were removed from v_i's list, and there is a color left to assign to v_i. Therefore this procedure results in a proper list coloring if $q \geq d + k + 1$, which shows we may simply output YES in such situations.

In the remaining cases, there are at most $d + k$ relevant color choices for a vertex. Hence we may apply standard dynamic programming techniques [16], since the number of states in a bag will not exceed $(k+1)^{d+k}$. As this term is bounded by a function of the parameter alone, the sketched approach solves the problem in FPT-time. □

This example shows how we may cope with the intractability of enriched problem models by distilling extra parameters from the hardness proofs. When applying this practice, one should take the problem domain into account to employ parameters which may reasonably be expected to be small on typical instances; something which is arguably lacking from the presented example.

Open Problem 10. Find a small parameter ℓ relating to the structure of the list function, such that List Coloring parameterized by treewidth plus ℓ is *FPT*.

As the coloring example illustrates, the goal of model enrichment naturally raises challenges to deconstruct the intractability that often arises from generalized problem statements. In the pursuit of practically relevant algorithmics, we have to find the right balance between the modeling power of our problems versus the amount of structure (in terms of combined parameters) which is needed to establish tractability. Such investigations therefore naturally turn into a dialectic between these two principles. A nice example of how this plays out for the problem Interval Constrained Coloring is given by Komusiewicz et al. [85], and similar case studies exist for other problems [5, 98, 100]. We end the section with a concrete challenge about model enrichment.

Open Problem 11. Find a meta-kernelization theorem (in the sense of [12, 60, 62]) which is applicable to problems on *weighted graphs*.

5.2. A Theory of Fully Multivariate Algorithmics

The principle of model enrichment leads naturally to a setting in which we simultaneously consider multiple parameters of a problem instance. Instead of performing a two-dimensional analysis, we would like to assess problem complexity through a *fully multivariate framework*. Unfortunately, the tools for such an

analysis are currently lacking. In a limited sense, the multivariate generalization of parameterized complexity theory is achieved by *aggregate parameterization*. The parameter k can encode a whole list of secondary measurements. For concreteness, let us illustrate by assuming k encodes (r, s, t), where these are three relevant secondary measurements. The qualitative distinction framed by parameterized complexity theory, allows us to determine whether the exponential costs (that we expect to be associated to an NP-hard problem with three such relevant secondary measurements), can be confined to *some function* $f(k)$ of these measurements. But we cannot distinguish, via aggregate parameterization, whether this function might be $f(k) = r^{s^t}$ or something more benign, such as $f(k) = 2^{\mathcal{O}(r)} + 2^{\mathcal{O}(s)} + 2^{\mathcal{O}(t)}$.

Presently, there is no satisfactory answer to the question of how parameterized complexity theory can be generalized beyond its successful two-dimensional outlook. It is natural to investigate this question, but there is of course no guarantee that there is any similarly natural and productive fully-multivariate generalization possible. In mathematics, complex analysis can be viewed as a productive "two-dimensional" generalization of real analysis, but it seems not to admit a really useful fully-multivariate generalization, despite some practical applications of quaternions. However, this is a different mathematical context, and the quest to generalize parameterized complexity and FPT to a fully-multivariate set of tools, is an ill-posed question. There can be a variety of ways to bring in several secondary measurements in interesting and relevant ways that go beyond simple *ad hoc* aggregate parameterization. Can we set up or conduct a useful and interesting completeness program based on contrasting trivariate function classes? This might be considered a minimalist objective, if generalization of parameterized complexity theory is possible.

6. Concluding Remarks

We have surveyed the driving ideas behind the parameter ecology program while highlighting technical challenges and recent results. Let us summarize the message that we have tried to convey through this lengthy discussion.

Real-life inputs to computational problems are not arbitrary, but inherit structural regularities and restrictions from the generative processes and underlying problem domains — this connection is coined the *ecology of computation*. Structure which is present in instances can be captured by suitable parameterizations, allowing an attack on the seeming intractability of practical problems. The parameterized complexity of a problem is determined by the *parameterization* rather than the underlying *classical* problem, and there are techniques available which can successfully be applied to many different problems under the same parameterization. Investigations of the complexity of various parameterizations should be backed by a thorough understanding of the interplay between these parameters, which can be obtained by organizing them in a hierarchy. To truly understand the nature of a problem's hardness, a game of *how low can you go* has to be played in this hierarchy to find the smallest parameterizations exhibiting tractability in the form of FPT-algorithms or polynomial kernels.

This calls for a race for the *best* parameterizations, which eventually reveals the boundaries of tractability.

To better serve the needs of practitioners, it is important to study whether positive results for abstracted, simple problems can be lifted to richer models which have a better fit to applications. Establishing tractability of such generalized problems for a single parameter might force us to move so far upwards in the parameter hierarchy that we cannot reasonably except practical instances to have small values for such parameters. In the spirit of *deconstructing intractability* we may opt alternatively for taking *additional* parameters instead of a single, larger parameter. Relevant secondary measurements to be used as parameters can be found by studying hardness proofs and asking: "Why are the produced instances *unreasonable*?". For a complete understanding of the way in which multiple parameters simultaneously influence the problem difficulty, a theory of *fully multivariate algorithmics* is necessary to refine the crude view offered by studying the complexity of aggregate parameterizations. Little is currently known about the potential for multidimensional generalizations of the parameterized framework. Interesting challenges await us.

References

[1] F. N. Abu-Khzam, M. R. Fellows, M. A. Langston, and W. H. Suters. Crown structures for vertex cover kernelization. *Theory Comput. Syst.*, 41(3):411–430, 2007. doi:10.1007/s00224-007-1328-0.

[2] J. Alber, M. R. Fellows, and R. Niedermeier. Polynomial-time data reduction for dominating set. *J. ACM*, 51(3):363–384, 2004. doi:10.1145/990308.990309.

[3] S. Arnborg, J. Lagergren, and D. Seese. Easy problems for tree-decomposable graphs. *J. Algorithms*, 12(2):308–340, 1991. doi:10.1016/0196-6774(91)90006-K.

[4] R. Bar-Yehuda and S. Even. A local-ratio theorem for approximating the weighted vertex cover problem. *Analysis and design of algorithms for combinatorial problems*, 109:27–45, 1985. doi:10.1016/S0304-0208(08)73101-3.

[5] N. Betzler, S. Hemmann, and R. Niedermeier. A multivariate complexity analysis of determining possible winners given incomplete votes. In *Proc. 21st IJCAI*, pages 53–58, 2009.

[6] S. Böcker, S. Briesemeister, Q. B. A. Bui, and A. Truß. Going weighted: Parameterized algorithms for cluster editing. *Theor. Comput. Sci.*, 410(52):5467–5480, 2009. doi:10.1016/j.tcs.2009.05.006.

[7] H. L. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM J. Comput.*, 25(6):1305–1317, 1996. doi:10.1145/167088.167161.

[8] H. L. Bodlaender. A partial k-arboretum of graphs with bounded treewidth. *Theor. Comput. Sci.*, 209(1-2):1–45, 1998. doi:10.1016/S0304-3975(97)00228-4.

[9] H. L. Bodlaender. Kernelization: New upper and lower bound techniques. In *Proc. 4th IWPEC*, pages 17–37, 2009. doi:10.1007/978-3-642-11269-0_2.

[10] H. L. Bodlaender, R. G. Downey, M. R. Fellows, and D. Hermelin. On problems without polynomial kernels. *J. Comput. Syst. Sci.*, 75(8):423–434, 2009. doi:10.1016/j.jcss.2009.04.001.

[11] H. L. Bodlaender, M. R. Fellows, and M. T. Hallett. Beyond NP-completeness for problems of bounded width (extended abstract): hardness for the W hierarchy. In *Proc. 26th STOC*, pages 449–458. ACM, 1994. doi:10.1145/195058.195229.

[12] H. L. Bodlaender, F. V. Fomin, D. Lokshtanov, E. Penninkx, S. Saurabh, and D. M. Thilikos. (Meta) Kernelization. In *Proc. 50th FOCS*, pages 629–638, 2009. doi:10.1109/FOCS.2009.46.

[13] H. L. Bodlaender, B. M. P. Jansen, and S. Kratsch. Cross-composition: A new technique for kernelization lower bounds. In *Proc. 28th STACS*, pages 165–176, 2011. doi:10.4230/LIPIcs.STACS.2011.165.

[14] H. L. Bodlaender, B. M. P. Jansen, and S. Kratsch. Kernel bounds for path and cycle problems. In *Proc. 6th IPEC*, pages 145–158, 2011. doi:10.1007/978-3-642-28050-4_12.

[15] H. L. Bodlaender, B. M. P. Jansen, and S. Kratsch. Preprocessing for treewidth: A combinatorial analysis through kernelization. In *Proc. 38th ICALP*, pages 437–448, 2011. doi:10.1007/978-3-642-22006-7_37.

[16] H. L. Bodlaender and A. M. C. A. Koster. Combinatorial optimization on graphs of bounded treewidth. *Comput. J.*, 51(3):255–269, 2008. doi:10.1093/comjnl/bxm037.

[17] H. L. Bodlaender and B. van Antwerpen-de Fluiter. Reduction algorithms for graphs of small treewidth. *Inf. Comput.*, 167(2):86–119, 2001. doi:10.1006/inco.2000.2958.

[18] A. Brandstädt, V. B. Le, and J. P. Spinrad. *Graph classes: a survey*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1999.

[19] J. F. Buss and J. Goldsmith. Nondeterminism within P. *SIAM J. Comput.*, 22(3):560–572, 1993. doi:10.1137/0222038.

[20] L. Cai. Parameterized complexity of vertex colouring. *Discrete Appl. Math.*, 127(3):415–429, 2003. doi:10.1016/S0166-218X(02)00242-1.

[21] L. Cai and D. W. Juedes. On the existence of subexponential parameterized algorithms. *J. Comput. Syst. Sci.*, 67(4):789–807, 2003. doi:10.1016/S0022-0000(03)00074-6.

[22] J. Chen, B. Chor, M. Fellows, X. Huang, D. W. Juedes, I. A. Kanj, and G. Xia. Tight lower bounds for certain parameterized NP-hard problems. *Inf. Comput.*, 201(2):216–231, 2005. doi:10.1016/j.ic.2005.05.001.

[23] J. Chen, Q. Feng, Y. Liu, S. Lu, and J. Wang. Improved deterministic algorithms for weighted matching and packing problems. *Theor. Comput. Sci.*, 412(23):2503–2512, 2011. doi:10.1016/j.tcs.2010.10.042.

[24] J. Chen, H. Fernau, I. A. Kanj, and G. Xia. Parametric duality and kernelization: Lower bounds and upper bounds on kernel size. *SIAM J. Comput.*, 37(4):1077–1106, 2007. doi:10.1137/050646354.

[25] J. Chen, I. A. Kanj, and W. Jia. Vertex cover: Further observations and further improvements. *J. Algorithms*, 41(2):280–301, 2001. doi:10.1006/jagm.2001.1186.

[26] J. Chen, I. A. Kanj, and G. Xia. Improved upper bounds for vertex cover. *Theor. Comput. Sci.*, 411(40-42):3736 – 3756, 2010. doi:10.1016/j.tcs.2010.06.026.

[27] M. Chlebík and J. Chlebíková. Crown reductions for the minimum weighted vertex cover problem. *Discrete Appl. Math.*, 156(3):292–312, 2008. doi:10.1016/j.dam.2007.03.026.

[28] B. Chor, M. Fellows, and D. W. Juedes. Linear kernels in linear time, or how to save k colors in $O(n^2)$ steps. In *Proc. 30th WG*, pages 257–269, 2004. doi:10.1007/978-3-540-30559-0_22.

[29] F. R. K. Chung. On the cutwidth and topological bandwidth of a tree. *SIAM J. Algebra. Discr.*, 6:268–277, 1985. doi:10.1137/0606026.

[30] F. R. K. Chung and P. D. Seymour. Graphs with small bandwidth and cutwidth. *Discrete Math.*, 75(1-3):113–119, 1989. doi:10.1016/0012-365X(89)90083-6.

[31] B. Courcelle. The monadic second-order logic of graphs I: Recognizable sets of finite graphs. *Inf. Comput.*, 85(1):12–75, 1990. doi:10.1016/0890-5401(90)90043-H.

[32] B. Courcelle, J. A. Makowsky, and U. Rotics. Linear time solvable optimization problems on graphs of bounded clique-width. *Theory Comput. Syst.*, 33(2):125–150, 2000. doi:10.1007/s002249910009.

[33] P. Crescenzi, R. Silvestri, and L. Trevisan. On weighted vs unweighted versions of combinatorial optimization problems. *Inf. Comput.*, 167(1):10–26, 2001. doi:10.1006/inco.2000.3011.

[34] M. Cygan, D. Lokshtanov, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. On the hardness of losing width. In *Proc. 6th IPEC*, pages 159–168, 2011. doi:10.1007/978-3-642-28050-4_13.

[35] M. Cygan, J. Nederlof, M. Pilipczuk, M. Pilipczuk, J. M. M. van Rooij, and J. O. Wojtaszczyk. Solving connectivity problems parameterized by treewidth in single exponential time. In *Proc. 52nd FOCS*, pages 150–159, 2011. doi:10.1109/FOCS.2011.23.

[36] M. Cygan, M. Pilipczuk, and M. Pilipczuk. On group feedback vertex set parameterized by the size of the cutset. *CoRR*, abs/1112.6255, 2011.

[37] M. Cygan, M. Pilipczuk, M. Pilipczuk, and J. O. Wojtaszczyk. On multiway cut parameterized above lower bounds. In *Proc. 6th IPEC*, pages 1–12, 2011. doi:10.1007/978-3-642-28050-4_1.

[38] B. de Fluiter. *Algorithms for Graphs of Small Treewidth*. PhD thesis, Utrecht University, 1997. Available from: http://igitur-archive.library.uu.nl/dissertations/01847381/inhoud.htm.

[39] E. DeLaViña and B. Waller. A note on a conjecture of Hansen et al. Manuscript. Available from: http://cms.dt.uh.edu/faculty/delavinae/research/DelavinaWaller2009.pdf.

[40] H. Dell and D. van Melkebeek. Satisfiability allows no nontrivial sparsification unless the polynomial-time hierarchy collapses. In *Proc. 42nd STOC*, pages 251–260, 2010. doi:10.1145/1806689.1806725.

[41] M. Dom, D. Lokshtanov, and S. Saurabh. Incompressibility through colors and IDs. In *Proc. 36th ICALP*, pages 378–389, 2009. doi:10.1007/978-3-642-02927-1_32.

[42] R. Downey, M. Fellows, B. Kapron, M. Hallett, and H. Wareham. The parameterized complexity of some problems in logic and linguistics. In *Proc. 3rd LFCS*, pages 89–100, 1994. doi:10.1007/3-540-58140-5_10.

[43] R. Downey and M. R. Fellows. *Parameterized Complexity*. Monographs in Computer Science. Springer, New York, 1999.

[44] R. G. Downey, V. Estivill-Castro, M. R. Fellows, E. Prieto, and F. A. Rosamond. Cutting up is hard to do: the parameterized complexity of k-cut and related problems. *Electr. Notes Theor. Comput. Sci.*, 78:209–222, 2003. doi:10.1016/S1571-0661(04)81014-4.

[45] R. G. Downey, M. R. Fellows, and M. A. Langston, editors. *The Computer Journal: Special Issue on Parameterized Complexity*, volume 51, 2008. doi:10.1093/comjnl/bxm111.

[46] R. G. Downey, M. R. Fellows, and U. Stege. Parameterized complexity: A framework for systematically confronting computational intractability. In *Contemporary Trends in Discrete Mathematics*, volume 49 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 49–99. AMS and DIMACS, 1999.

[47] R. G. Downey and D. M. Thilikos. Confronting intractability via parameters. *Computer Science Review*, 5(4):279–317, 2011. doi:10.1016/j.cosrev.2011.09.002.

[48] F. Eisenbrand and F. Grandoni. On the complexity of fixed parameter clique and dominating set. *Theor. Comput. Sci.*, 326(1-3):57–67, 2004. doi:10.1016/j.tcs.2004.05.009.

[49] V. Estivill-Castro, M. Fellows, M. Langston, and F. Rosamond. FPT is P-time extremal structure I. In *Proc. 1st ACiD*, pages 1–41, 2005.

[50] M. Fellows, F. Rosamond, and A. Slinko. Sensing God's will is fixed parameter tractable. Technical report, ResearchSpace@Auckland 561, 2008. Available from: http://hdl.handle.net/2292/5089.

[51] M. R. Fellows, F. V. Fomin, D. Lokshtanov, F. A. Rosamond, S. Saurabh, S. Szeider, and C. Thomassen. On the complexity of some colorful problems parameterized by treewidth. *Inf. Comput.*, 209(2):143–153, 2011. doi:10.1016/j.ic.2010.11.026.

[52] M. R. Fellows, D. Hermelin, and F. A. Rosamond. Well-quasi-orders in subclasses of bounded treewidth graphs. In *Proc. 4th IWPEC*, pages 149–160, 2009. doi:10.1007/978-3-642-11269-0_12.

[53] M. R. Fellows and M. A. Langston. Fast self-reduction algorithms for combinatorical problems of VLSI-design. In *Proc. 3rd AWOC*, pages 278–287, 1988. doi:10.1007/BFb0040395.

[54] M. R. Fellows and M. A. Langston. On well-partial-order theory and its application to combinatorial problems of VLSI design. *SIAM J. Discrete Math.*, 5(1):117–126, 1992. doi:10.1137/0405010.

[55] M. R. Fellows, D. Lokshtanov, N. Misra, M. Mnich, F. A. Rosamond, and S. Saurabh. The complexity ecology of parameters: An illustration using bounded max leaf number. *Theory Comput. Syst.*, 45(4):822–848, 2009. doi:10.1007/s00224-009-9167-9.

[56] M. R. Fellows, D. Lokshtanov, N. Misra, F. A. Rosamond, and S. Saurabh. Graph layout problems parameterized by vertex cover. In *Proc. 19th ISAAC*, pages 294–305, 2008. doi:10.1007/978-3-540-92182-0_28.

[57] Q. Feng, J. Wang, and J. Chen. Matching and P_2-packing: Weighted versions. In *Proc. 17th COCOON*, pages 343–353, 2011. doi:10.1007/978-3-642-22685-4_31.

[58] J. Fiala, P. A. Golovach, and J. Kratochvíl. Parameterized complexity of coloring problems: Treewidth versus vertex cover. *Theor. Comput. Sci.*, 412(23):2513–2523, 2011. doi:10.1016/j.tcs.2010.10.043.

[59] J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer-Verlag New York, Inc., 2006.

[60] F. Fomin, D. Lokshtanov, S. Saurabh, and D. M. Thilikos. Bidimensionality and kernels. In *Proc. 21st SODA*, pages 503–510, 2010.

[61] F. V. Fomin, S. Gaspers, P. A. Golovach, K. Suchan, S. Szeider, E. J. van Leeuwen, M. Vatshelle, and Y. Villanger. k-gap interval graphs. In *Proc. 10th LATIN*, pages 350–361, 2012. doi:10.1007/978-3-642-29344-3_30.

[62] F. V. Fomin, D. Lokshtanov, N. Misra, G. Philip, and S. Saurabh. Hitting forbidden minors: Approximation and kernelization. In *Proc. 28th STACS*, pages 189–200, 2011. doi:10.4230/LIPIcs.STACS.2011.189.

[63] F. V. Fomin, D. Lokshtanov, V. Raman, and S. Saurabh. Bidimensionality and EPTAS. In *Proc. 22nd SODA*, pages 748–759, 2011. Available from: http://arxiv.org/abs/1005.5449.

[64] A. Frank and E. Tardos. An application of simultaneous diophantine approximation in combinatorial optimization. *Combinatorica*, 7:49–65, 1987. doi:10.1007/BF02579200.

[65] M. Frick and M. Grohe. The complexity of first-order and monadic second-order logic revisited. *Ann. Pure Appl. Logic*, 130(1-3):3–31, 2004. doi:10.1016/j.apal.2004.01.007.

[66] R. Ganian. Twin-cover: Beyond vertex cover in parameterized algorithmics. In *Proc. 6th IPEC*, pages 259–271, 2011. doi:10.1007/978-3-642-28050-4_21.

[67] M. R. Garey and D. S. Johnson. *Computers and Intractability, A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, New York, 1979.

[68] M. Grohe, K. Kawarabayashi, D. Marx, and P. Wollan. Finding topological subgraphs is fixed-parameter tractable. In *Proc. 43rd STOC*, pages 479–488, 2011. doi:10.1145/1993636.1993700.

[69] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*, volume 2 of *Algorithms and Combinatorics*. Springer, 1988.

[70] J. Guo and R. Niedermeier. Invitation to data reduction and problem kernelization. *SIGACT News*, 38(1):31–45, 2007. doi:10.1145/1233481.1233493.

[71] G. Gutin, E. J. Kim, M. Lampis, and V. Mitsou. Vertex cover problem parameterized above and below tight bounds. *Theory Comput. Syst.*, 48(2):402–410, 2011. doi:10.1007/s00224-010-9262-y.

[72] F. Harary. *Graph theory*. Addison-Wesley series in mathematics. Perseus Books, 1994.

[73] P. Heggernes, P. van 't Hof, B. M. P. Jansen, S. Kratsch, and Y. Villanger. Parameterized complexity of vertex deletion into perfect graph classes. In *Proc. 18th FCT*, pages 240–251, 2011. doi:10.1007/978-3-642-22953-4_21.

[74] F. Henglein and H. G. Mairson. The complexity of type inference for higher-order typed lambda calculi. *J. Funct. Program.*, 4(4):435–477, 1994. doi:10.1017/S0956796800001143.

[75] R. Impagliazzo, R. Paturi, and F. Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001. doi:10.1006/jcss.2001.1774.

[76] B. Jansen. Kernelization for maximum leaf spanning tree with positive vertex weights. In *Proc. 7th CIAC*, pages 192–203, 2010. doi:10.1007/978-3-642-13073-1_18.

[77] B. M. P. Jansen and H. L. Bodlaender. Vertex cover kernelization revisited: Upper and lower bounds for a refined parameter. In *Proc. 28th STACS*, pages 177–188, 2011. doi:10.4230/LIPIcs.STACS.2011.177.

[78] B. M. P. Jansen and S. Kratsch. Data reduction for graph coloring problems. In *Proc. 18th FCT*, pages 90–101, 2011. doi:10.1007/978-3-642-22953-4_8.

[79] B. M. P. Jansen and S. Kratsch. On polynomial kernels for structural parameterizations of odd cycle transversal. In *Proc. 6th IPEC*, pages 132–144, 2011. doi:10.1007/978-3-642-28050-4_11.

[80] B. M. P. Jansen and S. Kratsch. On polynomial kernels for structural parameterizations of odd cycle transversal. *CoRR*, abs/1107.3658, 2011. Available from: http://arxiv.org/abs/1107.3658.

[81] R. Kannan. Minkowski's convex body theorem and integer programming. *Math. Oper. Res.*, 12:415–440, August 1987. doi:10.1287/moor.12.3.415.

[82] K. Kawarabayashi, B. Mohar, and B. A. Reed. A simpler linear time algorithm for embedding graphs into an arbitrary surface and the genus of graphs of bounded tree-width. In *Proc. 49th FOCS*, pages 771–780, 2008. doi:10.1109/FOCS.2008.53.

[83] S. Khuller. Algorithms column: the vertex cover problem. *SIGACT News*, 33(2):31–33, 2002. doi:10.1145/564585.564598.

[84] D. J. Kleitman and D. B. West. Spanning trees with many leaves. *SIAM J. Discret. Math.*, 4(1):99–106, 1991. doi:10.1137/0404010.

[85] C. Komusiewicz, R. Niedermeier, and J. Uhlmann. Deconstructing intractability - a multivariate complexity analysis of interval constrained coloring. *J. Discrete Algorithms*, 9(1):137–151, 2011. doi:10.1016/j.jda.2010.07.003.

[86] S. Kratsch and M. Wahlström. Representative sets and irrelevant vertices: New tools for kernelization. In *Proc. 53rd FOCS*, 2012. To appear. Available from: http://arxiv.org/abs/1111.2195.

[87] M. Lampis. A kernel of order $2k - c \log k$ for vertex cover. *Inf. Process. Lett.*, 111(23-24):1089–1091, 2011. doi:10.1016/j.ipl.2011.09.003.

[88] M. Lampis. Algorithmic meta-theorems for restrictions of treewidth. *Algorithmica*, 64(1):19–37, 2012. doi:10.1007/s00453-011-9554-x.

[89] O. Lichtenstein and A. Pnueli. Checking that finite state concurrent programs satisfy their linear specification. In *Proc. 12th POPL*, pages 97–107, 1985. doi:10.1145/318593.318622.

[90] D. Marx. Parameterized coloring problems on chordal graphs. *Theor. Comput. Sci.*, 351(3):407–424, 2006. doi:10.1016/j.tcs.2005.10.008.

[91] D. Marx. Parameterized complexity and approximation algorithms. *Comput. J.*, 51(1):60–78, 2008. doi:10.1093/comjnl/bxm048.

[92] D. Marx, B. O'Sullivan, and I. Razgon. Finding small separators in linear time via treewidth reduction. *CoRR*, abs/1110.4765, 2011. Available from: http://arxiv.org/abs/1110.4765.

[93] S. Mishra, V. Raman, S. Saurabh, and S. Sikdar. König deletion sets and vertex covers above the matching size. In *Proc. 19th ISAAC*, pages 836–847, 2008. doi:10.1007/978-3-540-92182-0_73.

[94] S. Mishra, V. Raman, S. Saurabh, S. Sikdar, and C. R. Subramanian. The complexity of finding subgraphs whose matching number equals the vertex cover number. In *Proc. 18th ISAAC*, pages 268–279, 2007. doi:10.1007/978-3-540-77120-3_25.

[95] S. Mishra, V. Raman, S. Saurabh, S. Sikdar, and C. R. Subramanian. The complexity of König subgraph problems and above-guarantee vertex cover. *Algorithmica*, 61(4):857–881, 2011. doi:10.1007/s00453-010-9412-2.

[96] G. Nemhauser and L. Trotter. Vertex packings: structural properties and algorithms. *Math. Program.*, 8:232–248, 1975. doi:10.1007/BF01580444.

[97] J. Nešetřil and S. Poljak. On the complexity of the subgraph problem. *Commentationes Mathematicae Universitatis Carolinae*, 26(2):415–419, 1985. Available from: http://dml.cz/dmlcz/106381.

[98] A. Nichterlein, M. Dom, and R. Niedermeier. Aspects of a multivariate complexity analysis for rectangle tiling. *Oper. Res. Lett.*, 39(5):346–351, 2011. doi:10.1016/j.orl.2011.06.001.

[99] R. Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press, 2006.

[100] R. Niedermeier. Reflections on multivariate algorithmics and problem parameterization. In *Proc. 27th STACS*, pages 17–32, 2010. doi:10.4230/LIPIcs.STACS.2010.2495.

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

[102] Parameterized complexity wiki: Table of FPT races. Available from: http://fpt.wikidot.com/fpt-races.

[103] L. Perkovic and B. A. Reed. An improved algorithm for finding tree decompositions of small width. *Int. J. Found. Comput. Sci.*, 11(3):365–371, 2000.

[104] M. Pilipczuk. Problems parameterized by treewidth tractable in single exponential time: A logical approach. In *Proc. 36th MFCS*, pages 520–531, 2011. doi:10.1007/978-3-642-22993-0_47.

[105] V. Raman, M. S. Ramanujan, and S. Saurabh. Paths, flowers and vertex cover. In *Proc. 19th ESA*, pages 382–393, 2011. doi:10.1007/978-3-642-23719-5_33.

[106] I. Razgon and B. O'Sullivan. Almost 2-SAT is fixed-parameter tractable. *J. Comput. Syst. Sci.*, 75(8):435–450, 2009. doi:10.1016/j.jcss.2009.04.002.

[107] N. Robertson and P. Seymour. Graph minors: A survey. In *Surveys in Combinatorics*, pages 153–171. Cambridge University Press, 1985.

[108] N. Robertson and P. D. Seymour. Graph minors. XX. Wagner's conjecture. *Journal of Combinatorial Theory, Series B*, 92(2):325–357, 2004. doi:10.1016/j.jctb.2004.08.001.

[109] R. Sasák. Comparing 17 graph parameters. Master's thesis, University of Bergen, 2010. Available from: http://hdl.handle.net/1956/4329.

[110] A. Schrijver. *Combinatorial Optimization. Polyhedra and Efficiency*. Springer, Berlin, 2003.

[111] P. D. Seymour and R. Thomas. Call routing and the ratcatcher. *Combinatorica*, 14(2):217–241, 1994. doi:10.1007/BF01215352.

[112] A. Soleimanfallah and A. Yeo. A kernel of order $2k - c$ for vertex cover. *Discrete Math.*, 311(10-11):892–895, 2011. doi:10.1016/j.disc.2011.02.014.

[113] S. Szeider. Not so easy problems for tree decomposable graphs. In *Selected and Revised Papers of ICDM 2008*, volume 13 of *India RMS Lecture Notes Series*, pages 179–190. Ramanujan Mathematical Society, 2010. Available from: http://arxiv.org/abs/1107.1177.

[114] S. Szeider. Monadic second order logic on graphs with local cardinality constraints. *ACM Trans. Comput. Logic*, 12:1–21, January 2011. doi:10.1145/1877714.1877718.

[115] J. Uhlmann and M. Weller. Two-layer planarization parameterized by feedback edge set. In *Proc. 7th TAMC*, pages 431–442, 2010. doi:10.1007/978-3-642-13562-0_39.

[116] J. M. M. van Rooij, H. L. Bodlaender, and P. Rossmanith. Dynamic programming on tree decompositions using generalised fast subset convolution. In *Proc. 17th ESA*, pages 566–577, 2009. doi:10.1007/978-3-642-04128-0_51.

[117] R. Williams. Improving exhaustive search implies superpolynomial lower bounds. In *Proc. 42nd STOC*, pages 231–240, 2010.

[118] R. Wilson. *Graphs, colourings and the four-colour theorem*. Oxford Science Publications, 2002.

[119] C.-K. Yap. Some consequences of non-uniform conditions on uniform classes. *Theor. Comput. Sci.*, 26:287–300, 1983. doi:10.1016/0304-3975(83)90020-8.