# Towards Fully Multivariate Algorithmics: Some New Results and Directions in Parameter Ecology

Michael Fellows

University of Newcastle, Callaghan, Australia
michael.fellows@newcastle.edu.au

**Abstract.** This paper reports on three recent research directions that begin to explore the subject of fully multivariate algorithmics, meaning by this the largely uncharted theoretical landscape that lies beyond parameterized complexity and algorithmics — itself a natural two-dimensional generalization of the familiar one-dimensional framework of $\mathcal{P}$ *versus* $\mathcal{NP}$.

## 1 Introduction

Parameterized complexity starts from the premise that there are usually secondary measurements, apart from the primary measurement of overall input 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 defining $\mathcal{FPT}$ and $\mathcal{XP}$, respectively.

This is formalized by saying that a *parameterized problem* $\Pi$ takes as input a pair $(x, k)$ where $k$ is the *parameter* (usually a positive integer). The problem $\Pi$ is *fixed-parameter tractable* 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 $k$. $\mathcal{XP}$ is the class of parameterized problems that are solvable in time $O(n^{g(k)})$.

Viewing complexities theories as driven by contrasting function classes, the classical framework of $\mathcal{P}$ *versus* $\mathcal{NP}$ is "about" the contrast between the univariate function classes: $\mathcal{P}$, solvability in time $O(n^c)$, and functions of the form $O(2^{n^c})$. Hardness for $\mathcal{NP}$ indicates that you will probably be stuck with a running time of the latter kind.

$\mathcal{FPT}$ generalizes $\mathcal{P}$, and parameterized complexity is a natural 2-dimensional sequel to the familiar one-dimensional classical framework. Of course, there may be more than one relevant measurement beyond the overall input size. So what is the natural 3-dimensional sequel? Could there be a mathematically elegant and useful *fully multivariate* framework? Presently, there are no satisfactory answers to these questions. In view of the theoretical and practical successes of parameterized complexity and algorithmics it is natural to investigate them. We report here on three recent research directions that begin to explore the possibilities for *fully multivariate algorithmics*.

J. Fiala, J. Kratochvíl, and M. Miller (Eds.): IWOCA 2009, LNCS 5874, pp. 2–10, 2009.

## 2 Background on Parameterized Complexity

Parameterized complexity analysis unfolds in the contrast between the "good class" of bivariate functions FPT, and the "bad class" of runtimes of the form $O(n^{g(k)})$ — solvability in such time defines the parameterized complexity class XP. To emphasize the contrast, one could also consider defining FPT *additively* as solvability in time $f(k) + n^c$. It turns out it makes no difference whether the parameterized complexity class FPT is defined multiplicatively or additively [13,10]. The basic issue in parameterized complexity is whether any exponential costs of the problem can be confined to the parameter.

### 2.1 Some Motivating Examples

Type Checking in ML.
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) [20], so the situation appears discouraging from the standpoint of classical complexity theory. However, the implementations work well in practice because the ML Type Checking problem is *FPT* with a running time of $O(2^k n)$, where $n$ is the size of the program and $k$ is the maximum nesting depth of the type declarations [22]. Since normally $k \leq 5$, the algorithm is clearly practical on the natural input distribution.

The central issue in parameterized complexity can also be illustrated by fundamental problems about graphs. Consider the following well-known problems:

Vertex Cover
*Input:* A graph $G = (V, E)$ and a positive integer $k$.
*Question:* Does $G$ have a vertex cover of size at most $k$? (A *vertex cover* is a set of vertices $V' \subseteq V$ such that for every edge $uv \in E$, $u \in V'$ or $v \in V'$ (or both).)

Dominating Set
*Input:* A graph $G = (V, E)$ and a positive integer $k$.
*Question:* Does $G$ have a dominating set of size at most $k$? (A *dominating set* is a set of vertices $V' \subseteq V$ such that $\forall u \in V$: $u \in N[v]$ for some $v \in V'$.)

Although 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 Vertex Cover that runs in time $O(2^k n)$
2. The best known algorithm for Dominating Set is only a minor improvement on the brute force algorithm of trying all $k$-subsets. For a graph on $n$ vertices this approach has a running time of $O(n^{k+1})$.

The table below shows the contrast between these two kinds of complexity.

In these two example problems, the parameter is the size of the solution being sought. But a parameter that affects the compexity of a problem can be many

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

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

things. The quest for FPT algorithms leads to a *positive toolkit* of methods for obtaining FPT algorithms. For the most up-to-date overview of such methods, see the surveys in [6].

Classically, evidence that a problem is unlikely to have an algorithm with a runtime in the good class 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 *negative toolkit*). The current tower of the main parameterized complexity classes is:

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

Parameterized by the size $k$ of a solution, the familiar Independent Set problem is complete for $W[1]$ and Dominating Set is complete for $W[2]$. The naturally parameterized Bandwidth problem is hard for $W[t]$ for all $t$ [2]. The best known algorithms for the parameterized Independent Set and Dominating Set problems are slight improvements on the brute-force approach of trying all $k$-subsets. The parameterized class $W[1]$ is strongly analogous to NP, because the $k$-Step Halting Problem for Turing machines of unlimited nondeterminism is complete for $W[1]$ [11]. FPT is equal to $M[1]$ if and only if the so-called *Exponential Time Hypothesis* fails [21,9]. There is an algorithm for the $k$-Independent Set problem that runs in time $O(n^{o(k)})$ if and only if FPT is equal to $M[1]$, and there is an algorithm for the $k$-Dominating Set problem that runs in time $O(n^{o(k)})$ if and only if FPT is equal to $M[2]$ [5].

For further background on parameterized complexity we refer the reader to the textbooks [10,15,23], and the double-special issue of surveys of aspects of the field and various application areas [6].

Relevant secondary measurements that affect problem complexity can be *many different things*, such as the size of the solution, aspects of the structure of typical instances, aspects of the algorithmic approach, or the goodness of an approximation. For any real-world problem, there may be several such secondary measurements in various ranges of magnitude that should be considered. Parameterized complexity establishes a qualitative framework where there is a place for only one such secondary measurement. It is quite natural to investigate the prospects for *fully multivariate frameworks* for qualitatively assessing computational complexity. We report here on three recent research directions that begin to explore the possibilities:

• *Parameter ecology*, where the two different measurements of input structure and solution size interact.
• *Algorithmic metatheorems* in the context of structural parameterizations.
• *Relationships between structural parameterizations.*

## 3 Parameter Ecology

Why *ecology*?

Let us consider again the example of Type Checking in ML. Although the problem is highly intractable from the classical point of view, the implemented ML compilers (that include type-checking subroutines) work efficiently. The explanation is that human-composed programs typically have a maximum type-declaration nesting depth of $k \leq 5$. The FPT type-checking subroutine that runs in time $O(2^k n)$ is thus entirely adequate in practice. The reason that naturally occuring programs have small nesting depth is because the programs would otherwise risk becoming incomprehensible to the programmer creating them!

What this example points to (we think) is that often the "inputs" to one computational problem 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.

It therefore seems to be useful to know how all the various parameterized structural notions interact with all the other computational objectives one might have. The familiar paradigm of efficiently solving various problems for graphs of bounded treewidth just represents one row of a matrix of algorithmic questions. Table 1 illustrates the idea. We use here the shorthand: TW is Treewidth, BW is Bandwidth, VC is Vertex Cover, DS is Dominating Set, G is Genus and ML is Max Leaf. The entry in the 2nd row and 4th column indicates that there is an *FPT* algorithm to optimally solve the Dominating Set problem for a graph $G$ of bandwidth at most $k$. The entry in the 4th 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. An entry in the table describes the current state of knowledge about the complexity of the problem where 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. The table just gives a few examples of the unbounded conceptual matrix that we are concerned with.

The complexity of graph problems, for graphs of bounded treewidth, is well-developed and supports many systematic approaches, such as described in [7,1,4,10,23,3]. For example, Minimum Coloring can be solved in time $O(n)$ for

**Table 2.** The Complexity Ecology of Parameters

| | TW | BW | VC | DS | G | ML |
|---|---|---|---|---|---|---|
| TW | *FPT* | $W[1]$-hard | *FPT* | *FPT* | ? | *FPT* |
| BW | *FPT* | $W[1]$-hard | *FPT* | *FPT* | ? | *FPT* |
| VC | *FPT* | *FPT* | *FPT* | *FPT* | ? | *FPT* |
| DS | $W[1]$-hard | ? | $W[1]$-hard | $W[1]$-hard | ? | ? |
| G | $W[1]$-hard | $W[1]$-hard | $W[1]$-hard | $W[1]$-hard | *FPT* | $W[1]$-hard |
| ML | *FPT* | *FPT* | *FPT* | *FPT* | *FPT* | *FPT* |

graphs of treewidth at most $k$. In the terminology of parameterized complexity Minimum Coloring is *fixed parameter tractable* for the parameter *treewidth*.

Highly structured graph problems lead to the consideration of structural parameters that are more restrictive than treewidth: other rows of the matrix. One might ask whether these rows are really interesting, since a graph of bounded *max leaf number* is severely restricted in its structure. To be fair, however, a graph of bounded treewidth is also severely restricted, in contrast to an arbitrary graph. How to determine whether a graph of bounded *max leaf number* is 3-colorable in the "best possible" FPT runtime is an easily stated problem for which the answer is not obvious. Another observation that points to the interest in these rows is that there are now known to be many examples of problems that are $W[1]$-hard parameterized by a bound on treewidth, including Bandwidth [2]; List Coloring, Pre-Coloring Extension and Equitable Coloring [14]; General Factor [26]; and Minimum Maximum Outdegree [27]. One must therefore look "below treewidth" for FPT structural parameterizations for these problems. Note that while both bounded *vertex cover number* and bounded *max leaf number* imply bounded treewidth, neither of these structural bounds implies a bound on the other. List Coloring even remains $W[1]$-hard for graphs of bounded *vertex cover number* [18]. Lastly, it seems that for severe structural parameterizations such as bounded *vertex cover number* and bounded *max leaf number*, different FPT techniques are brought forward to importance, such as well-quasi-ordering and bounded variable integer linear programming.

Some recent positive results include:

**Theorem 1.** *[18,19] The* Minimum Bandwidth *problem can be solved to optimality for graphs of bounded vertex cover number (bounded max leaf number) parameterized by the vertex cover number (max leaf number).*

**Open Problem.** The Optimal Linear Arrangement problem, that asks for an injective layout function $f : V \to N$ that minimizes $\sum_{u \neq v} |f(u) - f(v)|$, is open, parameterized by the vertex cover and max leaf numbers.

### 3.1 How to Parameterize?

There are two priciples of how to find relevant parameterizations that seem worthwhile to articulate, and that naturally lead in multivariate directions.

**Principle 1: Model Enrichment.** We have noted above that the graph MINIMUM COLORING problem is FPT when parameterized by the treewidth of a graph. Coloring models scheduling, but in the real world, it is often the case that not every timeblock (color) can be assigned to every task (because, for example, Bob may not be available for any meetings on Thursday). In the interests of greater realism and a better fit to applications, *we should always explore more richly structured variations of tractable problems.* Following this example, since MINIMUM COLORING is FPT parameterized by graph treewidth, it is worth exploring the complexity of MINIMUM LIST COLORING parameterized by treewidth, as this more structured problem has better traction on real applications. Unfortunately, in this case, we have the following negative result.

**Theorem 2.** *[14]* (MINIMUM) LIST COLORING *is hard for* $W[1]$*, parameterized by treewidth, and even when parameterized by vertex cover number.*

**Principle 2: Deconstruction of Intractability.** One way to find relevant parameterizations of hard problems is to examine the proofs of the intractability results, and ask, "Why are the instances produced in this argument *unreasonable*?" An examination of the negative result(s) for (MINIMUM) LIST COLORING in [14] shows that the reduction from MULTICOLOR CLIQUE produces instances of LIST COLORING where "most" vertices has lists of allowable colors of size 2. This is clearly unreasonable. Bob may not be available on Thursdays, but it seems unlikely that everybody is available only two hours per week! This in turn suggests richer parameterizations, such as treewidth, together with a bound on the average number of colors *not* on the list.

The interplay between these two principles of parameterization, inevitably leads to us to consider highly structured problems for parameters more restrictive than treewidth.

## 4 Algorithmic Metatheorems: Well-Quasiordering

One of the most powerful FPT classification tools is *well-quasiordering.* Graphs in general are well-quasiordered by minors (the celebrated Graph Minor Theorem), and also importantly, 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*) [24,25].

The study of general algorithmic machineries, such as well-quasiordering, for rows of the parameter ecology matrix other than bounded treewidth has scarcely begun. One recent positive result is described as follows.

By a *3-star path* we refer to the graph formed by adding a path linking two copies of $K(1,3)$. The resulting graph has two pairs of vertices of degree 1, each pair adjacent to a vertex of degree 3 (there are two of these) and all other vertices have degree 2. Let $\mathcal{U}$ be the set of graphs that consists of all the cycles, together with all of the *3-star paths.*

It is easy to check that $\mathcal{U}$ is an antichain in the induced subgraph order; it is in the sense of the following theorem a *universal antichain.*

**Theorem 3.** *[8,16,17]*
*(1) A family of graphs $\mathcal{F}$ that is a lower ideal in the (ordinary) subgraph order is well-quasiordered by the induced subgraph order, and has FPT order tests, if and only if $\mathcal{F} \cap \mathcal{U}$ is finite (where the order tests are parameterized by the size of the intersection).*
*(2) For any induced subgraph order antichain $\mathcal{U}'$ for which (1) holds, the symmetric difference $\Delta(\mathcal{U}, \mathcal{U}')$ is finite.*

As a corollary, graphs of bounded vertex cover number are well-quasiordered by induced subgraphs and have FPT order tests.

## 5 Relationships between Structural Parameters

Some rows of the parameter ecology matrix are *better* than others in a certain sense.

**Example: Topological Bandwidth.** The *topological bandwidth* of a graph is defined to be the minimum, taken over all subdivisions $G'$ of a graph $G$, of the bandwidth of $G'$. Determining whether a graph has topological bandwidth bounded by $k$, is hard for $W[t]$ for all $t$ [2]. However, the topological bandwidth of a graph is "sandwiched" by its cutwidth: $tbw(G) \leq cw(G)$ and $cw(G) \leq tbw(G)^2$, and furthermore, it is FPT to determine whether a graph has cutwidth at most $k$ [10]. Consequently, a family of graphs has *bounded cutwidth* if and only if it has *bounded topological bandwidth*, and the former is easier to determine.

These examples raise fundamental questions about general relationships between rows of the parameter ecology matrix. *When a row is hard to determine exactly (that is, NP-hard, or hard for $W[1]$) — when can we find a sandwiching structural parameter that is easier to determine?*

We currently know only one concrete example of when such "easier" structural sandwiching is (likely) impossible: the parameter is the *minimum independent domination number* of a graph $id(G)$ defined to be the minimum size of an independent dominating set of $G$.

**Theorem 4.** *[12] There is no FPT sandwiching parameter for $id(G)$ (for any computable sandwich bound) unless $FPT = W[2]$.*

## 6 Concluding Remark

We have surveyed some of the issues involved in the quest to develop *fully multivariate algorithmics* and a few recent research directions that lead us to consider how several different secondary measures with different roles interact in shaping the complexity of computational problems, and thereby the angles we have available to exploit in order to design useful algorithms for NP-hard general problems, in situations where we are attentive to natural input distributions. Much more remains to be explored.

**Acknowledgments.** Many thanks to Frances Rosamond for conversations and assistance with this paper, and the invited talk at IWOCA 2009 on which it is based.

## References

1. Arnborg, S., Lagergren, J., Seese, D.: Easy problems for tree-decomposable graphs. J. Algorithms 12, 308–340 (1991)
2. Bodlaender, H., Fellows, M., Hallett, M.: Beyond NP-completeness for problems of bounded width: hardness for the $W$ hierarchy. In: Proceedings of the ACM Symposium on the Theory of Computing (STOC), pp. 449–458 (1994)
3. Bodlaender, H.L., Koster, A.M.: Combinatorial optimisation on graphs of bounded treewidth. The Computer Journal 51, 255–269 (2007)
4. Bodlaender, H.L.: A linear time algorithm for finding tree-decompositions of small width. SIAM J. Computing 25, 1305–1317 (1996)
5. Chen, J., Chor, B., Fellows, M., Huang, X., Juedes, D., Kanj, I., Xia, G.: Tight lower bounds for certain parameterized NP-hard problems. Information and Computation 201, 216–231 (2005)
6. Fellows, M., Downey, R., Langston, M. (Guest eds.): The Computer Journal: Two special issues of surveys of various aspects of parameterized complexity and algorithmics. The Computer Journal 51(1&3) (2008)
7. Courcelle, B.: The monadic second order logic of graphs I: Recognizable sets of finite graphs. Information and Computation 85, 12–75 (1990)
8. Ding, G.: Subgraphs and well-quasi-ordering. J. Graph Theory 16, 489–502 (1992)
9. Downey, R., Estivill-Castro, V., Fellows, M., Prieto-Rodriguez, E., Rosamond, F.: Cutting up is hard to do: the parameterized complexity of $k$-cut and related problems. Electron. Notes Theor. Comp. Sci. 78, 205–218 (2003)
10. Downey, R.G., Fellows, M.R.: Parameterized Complexity. Springer, Heidelberg (1999)
11. Downey, R., Fellows, M., Hallett, M., Kapron, B., Wareham, H.T.: The parameterized complexity of some problems in logic and linguistics. In: Matiyasevich, Y.V., Nerode, A. (eds.) LFCS 1994. LNCS, vol. 813, pp. 89–100. Springer, Heidelberg (1994)
12. Parameterized approximation for dominating set problems. Information Processing Letters 109, 68–70 (2008)
13. Downey, R., Fellows, M., Stege, U.: Parameterized complexity: a framework for systematically confronting computational intractability. In: Graham, R., Kratochvil, J., Nesetril, J., Roberts, F. (eds.) Contemporary Trends in Discrete Mathematics, Proceedings of the DIMACS-DIMATIA Workshop on the Future of Discrete Mathematics, Prague, 1997. AMS-DIMACS Series in Discrete Mathematics and Theoretical Computer Science, vol. 49, pp. 49–99 (1999)
14. Fellows, M., Fomin, F., Lokshtanov, D., Rosamond, F., Saurabh, S., Szeider, S., Thomassen, C.: On the complexity of some colorful problems parameterized by treewidth. In: Dress, A.W.M., Xu, Y., Zhu, B. (eds.) COCOA 2007. LNCS, vol. 4616, pp. 366–377. Springer, Heidelberg (2007)
15. Flum, J., Grohe, M.: Parameterized Complexity Theory. Springer, Heidelberg (2006)
16. Fellows, M., Hermelin, D., Rosamond, F.: Well-quasi-ordering bounded treewidth graphs. In: Proceedings of IWPEC 2009. LNCS. Springer, Heidelberg (to appear,2009)

17. Fellows, M., Hermelin, D., Rosamond, F.: Parameter ecology, well-quasi-ordering and universal antichains (manuscript, 2009)
18. Fellows, M., Lokshtanov, D., Misra, N., Rosamond, F., Saurabh, S.: Graph Layout Problems Parameterized by Vertex Cover. In: Hong, S.-H., Nagamochi, H., Fukunaga, T. (eds.) ISAAC 2008. LNCS, vol. 5369, pp. 294–305. Springer, Heidelberg (2008)
19. Fellows, M., Lokshtanov, D., Misra, N., Mnich, M., Rosamond, F., Saurabh, S.: The complexity ecology of parameters: an illustration using bounded max leaf number. Theory of Computing Systems (2009)
20. Henglein, F., Mairson, H.G.: The complexity of type inference for higher-order typed lambda calculi. In: Proc. Symp. on Principles of Programming Languages (POPL), pp. 119–130. ACM Press, New York (1991)
21. Impagliazzo, R., Paturi, R.: Which problems have strongly exponential complexity? J. Computer and Systems Sciences 63, 512–530 (2001)
22. Lichtenstein, O., Pneuli, A.: Checking That Finite-State Concurrents Programs Satisfy Their Linear Specification. In: Proceedings of the 12th ACM Symposium on Principles of Programming Languages, pp. 97–107 (1985)
23. Niedermeier, R.: Invitation to Fixed Parameter Algorithms. Oxford University Press, Oxford (2006)
24. Robertson, N., Seymour, P.: Graph minors: a survey. In: Anderson, J. (ed.) Surveys in Combinatorics, pp. 153–171. Cambridge University Press, Cambridge (1985)
25. Robertson, N., Seymour, P.: Graph minors XX. Wagner's conjecture. J. Comb. Th. Series B 92, 325–357 (2004)
26. Szeider, S.: Monadic second order logic on graphs with local cardinality constraints. In: Ochmański, E., Tyszkiewicz, J. (eds.) MFCS 2008. LNCS, vol. 5162, pp. 601–612. Springer, Heidelberg (2008)
27. Szeider, S.: Not so easy problems for tree decomposable graphs. In: Proc. ICDM 2008 (2008) (to appear)