New Directions and New Challenges in Algorithm Design and Complexity, Parameterized

Michael R. Fellows

School of Electrical Engineering and Computer Science
University of Newcastle, University Drive, Callaghan NSW 2308, Australia
mfellows@cs.newcastle.edu.au

Abstract. The goals of this survey are to:
(1) Motivate the basic notions of parameterized complexity and give some examples to introduce the toolkits of FPT and W-hardness as concretely as possible for those who are new to these ideas.
(2) Describe some new research directions, new techniques and challenging open problems in this area.

1 Introduction

Let's immediately consider a compelling concrete motivation for the subject of parameterized complexity. Almost every computational problem that has been considered to date has turned out to be NP-hard (or worse). Vast efforts have subsequently been expended on the program of understanding the extent to which NP-hard problems might be approximated in polynomial time.

Thanks to the wonderful PCP Theorem [ALMSS92], it has been shown that:
• Large numbers, perhaps the majority, of the "classic" NP-hard optimization problems of computing cannot be approximated to within any constant factor, in polynomial time, unless $P = NP$.
• Of the remaining NP-hard problems, many do not admit *polynomial-time approximation schemes* (PTAS's) unless $P = NP$.
• But, "*good news*" (in a generally bleak landscape): dozens of hard problems do admit PTAS's. This means that for any fixed ϵ, there is an algorithm to compute a solution within a factor of $(1 + \epsilon)$ of optimal in polynomial time. (For a comprehensive account of approximation complexity see [ACGKMP99].) Upon examination, however, most of these PTAS results appear to be completely useless.

- The PTAS for the EUCLIDEAN TSP due to Arora [Ar96] has a running time of around $O(n^{300/\epsilon})$. Thus for a 20% error, we have a "polynomial-time" algorithm that runs in time $O(n^{1500})$.
- The PTAS for the MULTIPLE KNAPSACK problem due to Chekuri and Khanna [CK00] has a running time of $O(n^{12(\log(1/\epsilon)/\epsilon^8)})$. Thus for a 20% error we have a "polynomial-time" algorithm that runs in time $O(n^{9375000})$.

- The PTAS for the MAXIMUM SUBFOREST PROBLEM due to Shamir and Tsur [ST98] has a running time of $O(n^{2^{2^{1/\epsilon}-1}})$. For a 20% error we thus have a "polynomial" running time of $O(n^{958267391})$.
- The PTAS for the MAXIMUM INDENDENT SET problem on geometric graphs due to Erlebach, Jansen and Seidel [EJS01] has a running time of $O(n^{(4/\pi)(1/\epsilon^2+2)^2(1/\epsilon^2+1)^2})$. Thus for a 20% error, $O(n^{532804})$.
- The PTAS for the GENERAL MULTIPROCESSOR JOB SCHEDULING PROBLEM due to Chen and Miranda [CM99] runs in time $O(n^{(3mm!)^{(m/\epsilon)+1}})$ for m machines. Thus for 4 machines with a 20% error we have an algorithm that runs in time $O(n^{10000000000000000000000000000000000000000000000000000000000000000000000000000})$.

Most PTAS's are like that. The qualitative problem with the polynomial running times is that the parameter $k = 1/\epsilon$ is in the exponent. In some cases, an initial PTAS classification has subsequently been improved to one that does not have this problem. Arora described in 1997 an improved, nearly linear-time PTAS for the EUCLIDEAN TSP [Ar97]. A contemporary challenge is *to investigate which problems that have PTAS's admit PTAS's with any hope of practical significance, as exemplified by Arora's work on the Euclidean TSP.* The essential issue is how two measurements: n, the overall input size, and $k = 1/\epsilon$, the goodness-of-approximation parameter, qualitatively interact. The one-dimensional complexity framework of *P versus NP* is unable to speak to this. The two-dimensional parameterized complexity perspective provides the appropriate tools. The basic definitions are as follows.

Definition 1. *A parameterized language L is a subset $L \subseteq \Sigma^* \times \Sigma^*$. If L is a parameterized language and $(x, k) \in L$ then we will refer to x as the* main part*, and refer to k as the* parameter*.*

Definition 2. *A parameterized language L is fixed-parameter tractable (FPT) if it can be determined in time $f(k)q(n)$ whether $(x, k) \in L$, where $|x| = n$, $q(n)$ is a polynomial in n, and f is a function (unrestricted).*

The analogous additive "$f(k) + q(n)$" definition of FPT is equivalent.

Definition 3. *A parameterized language L belongs to the class XP if it can be determined in time $f(k)n^{g(k)}$ whether $(x, k) \in L$, where $|x| = n$, where f and g are unrestricted functions.*

It is known that FPT is (unconditionally) a proper subset of XP [DF99].

Examples of Parameterized Problems that are in XP.

(1) All of the problems that have PTAS's, where the parameter is $k = 1/\epsilon$.
(2) Many graph problems that take as input a graph G and a positive integer (parameter) k, such as determining whether a graph has a k-element dominating set, which is solvable in time $O(n^{k+1})$ by brute force checking of all k-subsets.
(3) The GRAPH ISOMORPHISM PROBLEM, parameterized by maximum vertex degree [Luks82].

(4) The μ-calculus model-checking problem, parameterized by formula size [EJ91].
(5) The problem of determining whether there is a relational structure homomorphism $h : A \to B$, parameterized by the hypertreewidth of A [GLS99].

Examples of Parameterized Problems that are FPT.

(1) The problem of approximating a solution to the Euclidean TSP problem, where the parameter is $k = 1/\epsilon$, the goodness of the approximation [Ar97].
(2) Many (maybe half) of the NP-hard graph problems that take as input a graph G and a positive integer parameter k, such as the VERTEX COVER problem, for which the current best FPT algorithm runs in time $O((1.29)^k + kn)$ [NR99,CKJ01]. Another nice example is the MAX LEAF SPANNING TREE problem. The trajectory of FPT algorithms for this problem includes:
• A quadratic FPT algorithm based on graph minors [FL92] having a completely ridiculous parameter function.
• The first linear time FPT algorithm for the problem due to Bodlaender [Bod93] had a parameter function of $f(k) = (17k^4)!$ or thereabouts.
• A linear time FPT algorithm by Downey-Fellows with $f(k) = (2k)^{4k}$ [DF95a].
• An improved linear time FPT algorithm due to Fellows, McCartin, Rosamond and Stege [FMRS00] with parameter function $f(k) = (14.23)^k$.
• The current best FPT algorithm for the problem, due to Bonsma, Brüggemann and Woeginger, running in time $O(n^3 + 9.4815^k k^3)$ [BBW03].
(3) The INTEGER LINEAR PROGRAMMING problem, where the parameter is the number of variable [Len83].
(4) The problem of determining whether a graph H is a minor of a graph G, where the parameter is H [RS95].
(5) The problem of determining whether there is a relational structure homomorphism $h : A \to B$, for structures of bounded arity, parameterized by the treewidth of the core of A [Gr03].

A parameter can be many things, and a single classical problem may be parameterized in a number of relevant ways. A parameter may be compound. The CLOSEST SUBSTRING problem is interestingly parameterized by the aggregate parameter consisting of: (1) the number of sequences, (2) the size of the alphabet, and (3) the Hamming distance from the input substrings to the motif [FGN03]. (Whether this parameterization of the problem is FPT is an open problem.)

In the next two sections we will survey, with some easy examples, the two toolkits that a working algorithmist might be interested in mastering. In the next section we survey the **bad news toolkit** of $W[1]$-hardness, the parameterized analog of the classical notion of NP-hardness that provides us with the means to show that some parameterized problems are unlikely to be in FPT. We will also discuss the new idea of $M[1]$-hardness, and its "lower bounds" applications. In §3 we will overview the **good news toolkit** of techniques for designing FPT algorithms. We will describe some striking new methods using vertex cover structure generally, and discuss the practical significance of FPT. In §4 we will summarize some of the main challenges for this area of research.

2 The Bad News Toolkit of $W[1]$ and $M[1]$ Hardness

The GRAPH k-CUT problem has a well-known $O(n^{k^2})$ algorithm due to Goldschmidt and Hochbaum [GH88,GH94] (also [KS96]).

GRAPH k-CUT
Input: A graph $G = (V, E)$, an edge-weighting function $w : E \to \mathbb{N}$, and positive integers k, m.
Parameter: k
Question: Is there a set $C \subseteq E$ with $\sum_{e \in C} w(e) \leq m$, such that the graph $G' = G - C$ obtained from G by removing the edges of the *cut* C has at least k connected components?

An $O(n^{k^2})$ algorithm is impractical for $k > 2$, and the same question arises as with the PTAS's. Before getting into the definitions that set up $W[1]$-hardness, let's look at a simple concrete example of the kind of problem reduction needed for the theory. (This example is taken from [DEFPR03].) Note that this reduction looks very similar to an NP-completeness reduction, and that it is *not difficult*.

Theorem 1. *If* GRAPH k-CUT *is FPT then so is the problem of determining whether a graph has a k-clique.*

Proof. We reduce the k-CLIQUE problem to GRAPH k-CUT. Let $G = (V, E)$ be the graph on n vertices for which we wish to determine if G has a k-clique. We construct a graph G' as follows:
(1) Start with $n + 2$ disjoint cliques of size n^4. We will refer to these as the *top clique* C_0, the *bottom clique* C_1 and the n *vertex cliques* C_v, $v \in V$.
(2) For each $v \in V$, connect C_v to C_1 by a matching of size $n^2 + n - deg(v)$.
(3) For each $v \in V$, connect C_v to C_0 by a matching of size $\binom{k}{2}$.
(4) For each edge $uv \in E$, make a single edge from C_u to C_v.
Each edge weight is 1; take $m = k(n^2 + n) + (k - 1)\binom{k}{2}$; take $k' = k + 1$.

We claim that G has a k-clique if and only if (G', k', m) is a yes-instance for the GRAPH k-CUT problem.

If G has a k-clique on the k vertices $V' \subseteq V$, then we can cut out k vertex cliques as follows (leaving the rest in a further component, noting $k' = k + 1$):
(1) To cut the k vertex cliques C_u, $u \in V'$, from C_1 requires almost all of the m cuts we are allowed. After we have accomplished this much, we have only $(k - 1)\binom{k}{2} + \sum_{u \in V'} deg(u)$ further cuts that we can make.
(2) The second term in the sum of our remaining budget of cuts from (1) allows us to cut the single edges joining the cliques corresponding to V' to the cliques corresponding to $V - V'$, and to each other (that is, the single edges between C_u and C_v where both $u, v \in V'$), requiring a number of cuts equal to all of this term, except for a savings of 1 for each adjacency of vertices in V'. Since there are $\binom{k}{2}$ such adjacencies, we have a remaining budget of $k\binom{k}{2}$ after this stage of cutting.
(3) The remaining budget of cuts allows us to cut all of the cliques corresponding to V' free from C_0.

The other direction is nearly obvious from the discussion above, and is easily checked.

To finish the argument, suppose that the GRAPH k-CUT problem has an FPT algorithm, that is, one that runs in time $f(k)n^c$ (where c is a constant and f is some function, unrestricted, as per the definition of FPT). The combinatorial reduction described above transforms an instance (G, k) of the k-CLIQUE problem to an instance (G', k') of the GRAPH k-CUT problem, where $k' = k+1$. But then we can solve k-CLIQUE in time $f(k+1)n^{c'}$ where $c' = 5c$.

A key point above is that the parameter k in the source problem is mapped to parameter k' of the target so that k' is purely a function of k.

Definition 4. *A parameterized language L is many:1 parametrically reducible to a parameterized language L' if there is an FPT algorithm that transforms (x, k) into (x', k') so that:*
(1) $(x, k) \in L$ if and only if $(x', k') \in L'$, and
(2) $k' = g(k)$ (for some function g).

A notion of problem reduction allows us to explore equivalence classes of parameterized problems. The main classes are organized in the hierarchy:

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

The maining working tool (so far) for showing likely intractability is $W[1]$-hardness, the parameterized analogue of NP-hardness. (The class $M[1]$ is something new, and we will say more about it below.) The k-CLIQUE problem is $W[1]$-complete [DF99], and is a frequent source for $W[1]$-hardness proofs. $W[1]$ is strongly analogous to NP because the k-STEP HALTING PROBLEM FOR NONDETERMINISTIC TURING MACHINES (with unlimited nondeterminism) is complete for $W[1]$ ([DF95b] + [CCDF97]).

2.1 How Do We Analyze PTAS's?

The following definition captures the essential issue.

Definition 5. *An optimization problem Π has an* efficient P-time approximation scheme *(EPTAS) if it can be approximated to a goodness of $(1+\epsilon)$ of optimal in time $f(k)n^c$ where c is a constant and $k = 1/\epsilon$.*

The following important connection was first proved by Bazgan, and later independently by Cesati and Trevisan [Baz95,CT97].

Theorem 2. *Suppose that Π_{opt} is an optimization problem, and that Π_{param} is the corresponding parameterized problem, where the parameter is the value of an optimal solution. Then Π_{param} is fixed-parameter tractable if Π_{opt} has an EPTAS.*

Here is an example of how Bazgan's Theorem can be used to analyze PTAS complexity. Khanna and Motwani introduced three planar logic problems in an effort to give a general explanation of PTAS-approximability. Their suggestion is that "hidden planar structure" in the logic of an optimization problem is what allows PTASs to be developed [KM96]. One of their three general planar logic optimization problems is defined as follows.

PLANAR TMIN
Input: A collection of Boolean formulas in sum-of-products form, with all literals positive, where the associated bipartite graph is planar (this graph has a vertex for each formula and a vertex for each variable, and an edge between two such vertices if the variable occurs in the formula).
Output: A truth assignment of minimum weight (i.e., a minimum number of variables set to *true*) that satisfies all the formulas.

By Bazgan's Theorem, it is enough to show that the problem of deciding whether there is a truth assignment of weight k, taking this as the parameter, is $W[1]$-hard.

Theorem 3. *Planar TMIN is hard for* $W[1]$ *and therefore does not have an EPTAS unless* $FPT = W[1]$.

The other two planar logic problems of Khanna and Motwani are also $W[1]$-hard [CFJR01]. The reductions are all from the parameterized CLIQUE problem and are comparable in difficulty to the example we have seen concerning the GRAPH k-CUT problem.

2.2 A Surprising New Development

A natural (and useful!) class of parameterized problems that may be intermediate between FPT and $W[1]$ has recently been identified. The situation is:

$$FPT \subseteq M[1] \subseteq W[1]$$

The combinatorics of $M[1]$ are rich and easy to work with. It may actually be a better primary reference point for intractability than $W[1]$. There are two natural "routes" to $M[1]$.
• First of all, there are FPT algorithms for many parameterized problems (e.g., VERTEX COVER) that run in time $2^{O(k)}n^c$. VERTEX COVER has a simple $2^k n$ FPT algorithm that we can consider here as an example. We now "renormalize" and define the problem:

$k \log n$ VERTEX COVER
Input: A graph G on n vertices and a positive integer k.
Parameter: k
Question: Does G have a vertex cover of size at most $k \log n$?

The simple FPT algorithm for the original VERTEX COVER problem, parameterized by the number of vertices in the vertex cover, allows us to solve this new problem in time $O(n^{k+1})$. It now makes sense to ask if the $k \log n$ VERTEX

COVER problem is in FPT — or is it $W[1]$-hard? It turns out that $k \log n$ VERTEX COVER is $M[1]$-complete. Call this the *renormalization route* to $M[1]$.

• There is another route to $M[1]$ via *miniaturization*. We certainly know an algorithm to solve n-variable 3SAT in time $O(2^n)$. So consider the following parameterized problem.

MINI-3SAT
Input: Positive integers k and n in unary, and a 3SAT expression E having at most $k \log n$ variables.
Parameter: k
Question: Is E satisfiable?

Using our exponential time algorithm for 3SAT, MINI-3SAT is in XP and we can wonder where it belongs — is it is FPT or $W[1]$-hard? This problem also turns out to be complete for $M[1]$.

Dozens of renormalized FPT problems and miniaturized arbitrary problems are now known to be $M[1]$-complete [DFPR03]. However, what is known is quite problem-specific. For example, one might expect MINI-MAX LEAF to be $M[1]$-complete, but all that is known presently is that it is $M[1]$-hard. It is not known to be $W[1]$-hard, nor is it known to belong to $W[1]$.

The following theorem would be interpreted by most people as indicating that probably $FPT \neq M[1]$. (The theorem is essentially due to Cai and Juedes [CJ01].)

Theorem 4. *$FPT = M[1]$ if and only if n-variable 3SAT can be solved in time $2^{o(n)}$.*

Importantly, it is pretty easy to use $M[1]$ for various purposes. It supports convenient although unusual combinatorics. For example, one of the problems that is $M[1]$-complete is the miniature of the INDEPENDENT SET problem defined as follows.

MINI-INDEPENDENT SET
Input: Positive integers k and n in unary, a positive integer $r \leq n$, and a graph G having at most $k \log n$ vertices.
Parameter: k
Question: Does G have an independent set of size at least r?

The following example of some of the combinatorial possibilities available in making reductions from an $M[1]$-hard problem is due to [DEFPR03].

Theorem 5. *There is an FPT reduction from* MINI-INDEPENDENT SET *to ordinary parameterized* INDEPENDENT SET *(parameterized by the number of vertices in the independent set).*

Proof. Let $G = (V, E)$ be the miniature, for which we wish to determine whether G has an independent set of size r. Here, of course, $|V| \leq k \log n$ and we may regard the vertices of G as organized in k blocks $V_1, ..., V_k$ of size $\log n$. We now employ a simple but useful *counting trick* that can be used when reducing

miniatures to "normal" parameterized problems. Our reduction is a Turing reduction, with one branch for each possible way of writing r as a sum of k terms, $r = r_1 + \cdots + r_k$, where each r_i is bounded by $\log n$. The reader can verify that $(\log n)^k$ is an FPT function, and thus that there are an allowed number of branches. A branch represents a commitment to choose r_i vertices from block V_i (for each i) to be in the independent set.

We now produce (for a given branch of the Turing reduction) a graph G' that has an independent set of size k if and only if the miniature G has an independent set of size r, distributed as indicated by the commitment made on that branch. The graph G' consists of k cliques, together with some edges between these cliques. The ith clique consists of vertices in 1:1 correspondence with the subsets of V_i of size r_i. An edge connects a vertex x in the ith clique and a vertex y in the jth clique if and only if there is a vertex u in the subset $S_x \subseteq V_i$ represented by x, and a vertex v in the subset $S_y \subseteq V_j$ represented by y, such that $uv \in E$.

Verification that the reduction works correctly is straightforward.

The hypothesis that $FPT \neq M[1]$ allows us to say some very interesting things. Chor, Fellows and Juedes have recently proved the following using arguments similar to the proof of Theorem 5.

Theorem 6. *If $FPT \neq M[1]$ then determining whether a graph has a k-element dominating set (k-element independent set) cannot be solved in time $O(n^{o(k)})$.*

Cai and Juedes [CJ01] proved the following path-breaking results, which established an **FPT optimality program** of broad applicability.

Theorem 7. *If $FPT \neq M[1]$ then there cannot be an FPT algorithm for the general* Vertex Cover *problem with a parameter function of the form $f(k) = 2^{o(k)}$, and there cannot be an FPT algorithm for the* Planar Vertex Cover *problem with a parameter function of the form $f(k) = 2^{o(\sqrt{k})}$.*

3 The Good News Toolkit of FPT Methods

The best available general reference regarding FPT methods is Rolf Niedermeier's recent Habilitationschrift [Nie02], that we can only hope will be expanded into a comprehensive monograph. In a few pages, we can only point to some of the range and mathematical depth of this area, and sample a few key ideas and recent developments.

Some of the methods in the FPT toolkit, such as **well-quasi-ordering and graph minors** depend on very deep results that provide what are best regarded as *FPT classification tools*. These are powerful and general, but because of that generality, unlikely to yield directly useful FPT algorithms. The FPT classification just gives a starting point — recall the trajectory of improved FPT algorithms for Max Leaf discussed in §1.

Such classification tools can be quite easy to use. For example, to *use* the graph minors well-quasiordering (and accompanying GRAPH MINOR CONTAINMENT FPT result) of Robertson and Seymour [RS95] (based on nearly a thousand journal pages of intricate argument) to conclude that GRAPH GENUS is FPT — one only needs to check that, for every fixed k, the family of graphs that have genus at most k, is closed under the local operations that define the minor order: (1) deleting an edge, (2) deleting an isolated vertex, and (3) contracting an edge. The results of Robertson and Seymour on graph minors were observed to initially classify VERTEX COVER, MAX LEAF and many other problems as FPT in this way [FL92].

A less expensive alternative for these classifications and some others (via well-quasiordering methodology) is established by the following theorem [AFLR03].

Theorem 8. *For every fixed k, the family of graphs having a vertex cover of size at most k is well-quasiordered by ordinary subgraphs. Furthermore, restricting to this family of graphs, the* SUBGRAPH *problem (Is H a subgraph of G?), parameterized by H, is linear time FPT.*

The proof of this theorem requires less than two pages. It is more general and easier to use (than graph minors) — on those parameterized problems that are subject to *bounded vertex cover structure*. For example, VERTEX COVER could be classified as FPT by the following algorithm:
(1) Compute a maximal matching by a greedy procedure. If the matching has size greater than k then the answer is NO.
(2) If the size of the matching is at most k, then we have a $2k$ vertex cover, and can use Theorem 8, noting that the NO instances (for fixed k) are closed in the subgraph order.

The same argument works to classify as FPT the problem MAX INTERNAL of determining whether a graph has a spanning tree with at least k internal vertices — although in case (1) the answer is YES rather than NO (because an easy induction shows that if a connected graph has a k-matching, then it is a YES instance for k-MAX INTERNAL). A better FPT algorithm for this problem is described elsewhere in these proceedings [PS03].

Duality. It is an important fact that it makes sense to parameterize "upward" as well as "downward" for NP-hard problems. Note that a graph G has a k-element independent set, if and only if G has a vertex cover of size $n-k$, if and only if the complement of G has a k-clique. The naturally parameterized INDEPENDENT SET (and CLIQUE) problems are thus *parametrically dual* in a sense whose importance was first recognized by Raman and Mahajan [MR99,KR00]. They noted the remarkable (but not universal) phenomenon that for a great many NP-hard problems, one of a dual pair will be $W[1]$-hard, and the other will be FPT.

Many of the exciting new applications in computational biology of the hugely successful new parallel implementations of the current best FPT VERTEX COVER algorithms of Dehne [DRST02], and Langston, et al. [ALSS03], *are really applications of* CLIQUE*, in various data clustering contexts, that are being solved*

"through the back door" by means of the FPT vertex cover algorithm. Initial results for their joint implementations suggest practicality for reduced graphs (not subject to further kernelization) for k up to something like 2,000. This means that for $n \leq 2,000$, the CLIQUE problem can be solved exactly by deleting a minimum vertex cover from the dual. The "method" of **parametric duality** is this: if a problem is $W[1]$-hard, consider the dual!

Taking our own advice, we note that DOMINATING SET is $W[2]$-complete [DF99], and that GRAPH COLORING (parameterized by the number of colors) is hard for $W[P]$ [DFS99]. The parametric duals of both of these problems are easily classified as FPT by using Theorem 8. (The graph minor results do not apply to either of these problems.)

Another powerful and deep FPT classification technique is the **method of color-coding** introduced by Alon, Yuster and Zwick [AYZ95], and exposited also in [DF99,Nie02]. This is mathematically interesting, although not yet of direct practical importance.

The FPT method of **bounded variable integer linear programming** [Len83] has been used to classify a number of problems as FPT for which no other FPT algorithm is known. The method is exposited in [Nie02] with some important examples. The practicality of this FPT methodology also has yet to be determined.

3.1 The Universal Parameter of Treewidth

The FPT algorithmics of **bounded treewidth** seems to be of universal applicability (cf. [GLS99,Gr03]), although Bodlaender's algorithm [Bod96] for producing a tree-decomposition of width at most k (if one exists) that runs in time $O(2^{35k^3}n)$ is not practical for very large values of k.

However, Bodlaender's algorithm for bounded treewidth is not always necessary. Sometimes, one can in polynomial-time *either* determine the answer, *or* get a bounded tree or path-width decomposition "for free". A nice example of this is described in [PS03].

3.2 Crown Rules: A Surprising New Kind of Kernelization Method

A surprisingly general new approach to kernelization rules has recently developed. Consider the VERTEX COVER problem, and the reduction rule for this problem for vertices of degree 1. If (G, k) is an instance to the VERTEX COVER problem where G has a vertex v of degree 1 with neighbor $u \neq v$, then we can reduce to the instance (G', k') where $G' = G - u - v$ and $k' = k - 1$. This simple *local* reduction rule admits a global structural generalization.

Definition 6. *A* crown decomposition *of G is a partition of the vertex set of G into three sets, H, C and J such that the following conditions hold:*
(1) C is an independent set,
(2) H is matched into C, and
(3) H is a cutset, i.e., there are no edges between C and J.

If (G, k) is an instance of the VERTEX COVER problem, and G admits a crown decomposition, then we can reduce to (G', k') where $G' = G - H - C$ and $k' = k - |H|$, as a generalization of the degree 1 reduction rule. Presently, it is an open problem whether determining if a graph has a crown decomposition is in P, or is NP-complete. However, if we are given: (1) a graph $G = (V, E)$ without isolated vertices, and (2) a vertex cover V' for G that satifies the inequality $2|V'| < |V|$ (in other words, the vertex cover is less than half the size of G), then we can compute a crown decomposition in polynomial time as follows. Let $V'' = V - V'$, and compute a maximum matching between V' and V''. For $x \in V'$, let $m(x) \in V''$ denote the vertex of V'' (if any) to which x is matched. For $U \subseteq V'$, define $m(U) = \{v \in V'' : \exists x \in U \text{ with } m(x) = v\}$. For $W \subseteq V''$ let $N(W) = \{x \in V' : \exists y \in W, xy \in E\}$. Let $A = V'' - m(V')$. The size condition on the vertex cover V' insures that A is nonempty. Note that since V' is a vertex cover, V'' is an independent set in G.

If we now take $C = m(N(A)) \cup A$, $H = N(A)$, and $J = V - C - H$, then it is easy to verify (using the fact that m is a *maximum* matching) that we have a nontrivial crown decomposition of G — and thus any instance (G, k) of the VERTEX COVER problem can be reduced by our *crown reduction rule*. It seems that many parameterized problems that are "subject to vertex cover structure" have reduction rules that are flavors of this approach. This includes many of the problems mentioned in this section.

Data reduction and kernelization rules are one of the primary outcomes of research on parameterized complexity. After all, whether k is small or not — and no matter if you are going to do simulated annealing eventually — it *always* makes sense to simplify and reduce (pre-process) your input! This humble activity, because reduction rules frequently cascade, can sometimes lead to unexpected success against hard problems on real input distributions. A fine example of this has been described by Weihe [Wei98].

4 Major Challenges

These seem to be some of the main challenges that lie ahead in this area of research.

Challenge 1: Explore the parameterized complexity of the fundamental problems of computer science subdiscipline X. Possible application are all across computer science and many are still completely unexplored.

Challenge 2: Clarify the complexity of PTAS's. This is a big project, with many dozens of problems with PTAS's unresolved. Only a handful of $W[1]$-hardness results are so far known [CT97,CFJR01,GGN03].

Challenge 3: Show that $M[1] = W[1]$. This is arguably the most important unsolved problem in the foundations of parameterized complexity. Showing this equality would join the parameterized and classical worlds at the root in a very elegant way. Intuitively, this challenge involves reducing CLIQUE (where the parameter k is the size of the "small" clique) to MINI-CLIQUE, where the graph

has been miniaturized, but the clique we are looking for may be "as big" as the graph. The coding that is required seems reminiscent of the holographic issues that arise in the proof of the PCP Theorem.

Challenge 4: Use the unusual but supple combinatorics of $M[1]$-hardness to expand the realm of parameterized intractability results. It may be that $M[1]$ gives us the means to show that GRAPH ISOMORPHISM for the parameter maximum degree is unlikely to be in FPT (and, in particular, therefore unlikely to be in P). Will it turn out that $M[1]$-hardness is *generally* easier (or more powerful) to work with than $W[1]$-hardness?

Challenge 5: Investigate the known FPT algorithms according to the optimality program of Cai and Juedes. For example, Bodlaender's FPT algorithm for TREEWIDTH has an exponential function of the form $f(k) = 2^{O(k^3)}$. We should therefore seek either a lower bound result that says this cannot be improved to $f(k) = 2^{o(k^3)}$ unless $FPT = M[1]$, or an improved FPT algorithm accompanied by a matching lower bound.

Challenge 6: Explore whether a lower bound program concerning P-time kernelization can be developed along similar lines into a routine methodology.

Challenge 7: Understand the broader opportunities for *crown rules* and other global structure problem reduction rules of this type.

Challenge 8: Explore the parameterized complexity of local search. For a concrete example of this, the k-change neighborhood of a tour for the TRAVELING SALESMAN PROBLEM can be explored for an improved solution in time $O(n^{2k+1})$ by brute force. Is this problem in FPT, or is it hard for $W[1]$? A moment's reflection shows that we can ask similar questions for almost every local search problem. Nothing much is currently known about this universally relevant parameterization of a stalwart method of practical algorithmics.

Challenge 9: Explore the possibility of cryptosystems whose security rests on the unlikeliness that some problem is in FPT. The cryptosystem would be parameterized, and for every fixed parameter value, crackable in polynomial time. Such possibilities have attracted very little investigation in the reigning paradigm of one-dimensional complexity.

Challenge 10: Develop the structure theory required to settle some of the major unresolved concrete complexity problems. If Robertson and Seymour had set out to prove that the GRAPH MINOR CONTAINMENT problem (determining whether H is a minor of G, parameterized by H) is FPT, they would *still* have had to develop much of the deep and beautiful structure theory of minors. One wonders whether some of the notable open problems of parameterized complexity require similar structure theory to be resolved. A prominent example is the problem of determining whether it is possible to delete k vertices from a directed graph so that the result is acyclic. (An FPT algorithm for this would have important applications in computational biology [Hal03].)

References

[ACGKMP99] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela and M. Protasi. *Complexity and Approximation.* Springer-Verlag, 1999.

[AFN02] J. Alber, M. Fellows and R. Niedermeier. "Efficient Data Reduction for Dominating Set: A Linear Problem Kernel for the Planar Case." *Proceedings of Scandinavian Workshop on Algorithms and Theory* (SWAT 2002), Springer-Verlag, *Lecture Notes in Computer Science* 2368 (2002), 150–159.

[AFLR03] F. Abu-Khzam, M. Fellows, M. Langston and F. Rosamond. "Bounded Vertex Cover Structure and the Design of FPT Algorithms." Manuscript, 2003.

[ALSS03] F. Abu-Khzam, M. A. Langston, P. Shanbhag and C. T. Symons. "High Performance Tools for Fixed-Parameter Tractable Implementations." WADS'03 Workshop Presentation, 2003.

[ALMSS92] S. Arora, C. Lund, R. Motwani, M. Sudan and M. Szegedy. Prof verification and intractability of optimization problems. *Proceedings of the IEEE Symposium on the Foundations of Computer Science* (1992).

[Ar96] S. Arora. "Polynomial Time Approximation Schemes for Euclidean TSP and Other Geometric Problems." In: *Proceedings of the 37th IEEE Symposium on Foundations of Computer Science* (1996), 2–12.

[Ar97] S. Arora. "Nearly Linear Time Approximation Schemes for Euclidean TSP and Other Geometric Problems." *Proc. 38th Annual IEEE Symposium on the Foundations of Computing* (FOCS'97), IEEE Press (1997), 554-563.

[AYZ95] N. Alon, R. Yuster and U. Zwick. "Color-coding." *Journal of the ACM* 42 (1995), 844–856.

[Baz95] C. Bazgan, "Schémas d'approximation et complexité paramétrée." Rapport de stage de DEA d'Informatique à Orsay, 1995.

[BBW03] P.S. Bonsma, T. Brüggemann and G.J. Woeginger. A faster FPT algorithm for finding spanning trees with many leaves. Manuscript, 2003.

[Bod93] H. L. Bodlaender. "On Linear Time Minor Tests and Depth-First Search." *Journal of Algorithms* 14 (1993), 1–23.

[Bod96] H. L. Bodlaender. "A Linear Time Algorithm for Finding Tree-Decompositions of Small Treewidth." *SIAM Journal on Computing* 25 (1996), 1305–1317.

[CCDF97] L. Cai, J. Chen, R. Downey and M. Fellows. "On the Parameterized Complexity of Short Computation and Factorization." *Archive for Mathematical Logic* 36 (1997), 321–337.

[CFJR01] Liming Cai, M. Fellows, D. Juedes and F. Rosamond, "Efficient Polynomial-Time Approximation Schemes for Problems on Planar Structures: Upper and Lower Bounds." Manuscript, 2001.

[CJ01] L. Cai and D. Juedes. "On the Existence of Subexponential Parameterized Algorithms," manuscript, 2001. Revised version of the paper, "Subexponential Parameterized Algorithms Collapse the W-Hierarchy," in: *Proceedings 28th ICALP*, Springer-Verlag LNCS 2076 (2001), 273–284. (The conference version contains some major flaws.)

[CK00] C. Chekuri and S. Khanna. "A PTAS for the Multiple Knapsack Problem." *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms* (SODA 2000), pp. 213-222.

[CKJ01] J. Chen, I. A. Kanj and W. Jia. "Vertex Cover: Further Observations and Further Improvements." *Journal of Algorithms* 41 (2001), 280–301.

[CM99] J. Chen and A. Miranda. "A Polynomial-Time Approximation Scheme for General Multiprocessor Scheduling." *Proc. ACM Symposium on Theory of Computing* (STOC '99), ACM Press (1999), 418–427.

[CT97] M. Cesati and L. Trevisan. "On the Efficiency of Polynomial Time Approximation Schemes." *Information Processing Letters* 64 (1997), 165–171.

[DEFPR03] R. Downey, V. Estivill-Castro, M. Fellows, E. Prieto-Rodriguez and F. Rosamond. "Cutting Up is Hard to Do: the Parameterized Complexity of k-Cut and Related Problems." *Electronic Notes in Theoretical Computer Science* 78 (2003), 205–218.

[DF95a] R. G. Downey and M. R. Fellows. "Parameterized Computational Feasibility." In: *P. Clote and J. Remmel (eds.), Feasible Mathematics II.* Birkhauser, Boston (1995), 219–244.

[DF95b] R. G. Downey and M. R. Fellows. "Fixed Parameter Tractability and Completeness II: Completeness for $W[1]$." *Theoretical Computer Science* A 141 (1995), 109–131.

[DF99] R. G. Downey and M. R. Fellows. *Parameterized Complexity.* Springer-Verlag, 1999.

[DFPR03] R. Downey, M. Fellows, E. Prieto-Rodriguez and F. Rosamond. "Fixed-parameter Tractability and Completeness V: Parametric Miniatures." Manuscript, 2003.

[DFS99] R. Downey, M. Fellows and U. Stege. "Parameterized Complexity: A Framework for Systematically Confronting Computational Intractability." In: *Contemporary Trends in Discrete Mathematics*, (R. Graham, J. Kratochvíl, J. Nestríl and F. Roberts, eds.), Proceedings of the DIMACS-DIMATIA Workshop, Prague, 1997, *AMS-DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, vol. 49 (1999), 49–99.

[Dow03] R. Downey. "Parameterized Complexity for the Skeptic." *Proceedings of the IEEE Computational Complexity Conference* (CCC'03), IEEE Press (2003), to appear.

[DRST02] F. Dehne, A. Rau-Chaplin, U. Stege and P. Taillon. "Solving Large FPT Problems on Coarse Grained Parallel Machines." Manuscript, 2002.

[EJ91] E.A. Emerson and C.S. Jutla. "Tree Automata, μ-Calculus and Determinacy." *Proceedings of the IEEE Symposium on Foundations of Computer Science (FOCS'91)*, IEEE Press (1991), 368–377.

[EJS01] T. Erlebach, K. Jansen and E. Seidel. "Polynomial Time Approximation Schemes for Geometric Graphs." *Proc. ACM Symposium on Discrete Algorithms (SODA'01)*, ACM Press (2001), 671–679.

[FGN03] M. Fellows, J. Gramm and R. Niedermeier. "On the Parameterized Intractability of Motif Search Problems." Manuscript, 2003.

[FL92] M.R. Fellows and M.A. Langston. "On Well-Partial-Order Theory and its Applications to Combinatorial Problems of VLSI Design." *SIAM Journal on Discrete Mathematics* 5, 117–126.

[FMRS00] M. Fellows, C. McCartin, F. Rosamond and U. Stege. "Coordinatized Kernels and Catalytic Reductions: An Improved FPT Algorithm for Max Leaf Spanning Tree and Other Problems." *Proceedings of the 20th Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'2000)*, Springer, Lecture Notes in Computer Science 1974 (2000), 240-251.

[GGN03] J. Gramm, J. Guo and R. Niedermeier. "On Exact and Approximation Algorithms for Distinguishing Substring Selection." To appear in the proceedings of FCT'03.

[GH88] O. Goldschmidt and D.S. Hochbaum. "A Polynomial Algorithm for the k-Cut Problem." *Proc. 29th Annual Symp. on the Foundations of Computer Science (FOCS)* (1988), 444-451.

[GH94] O. Goldschmidt and D.S. Hochbaum. "A Polynomial Algorithm for the k-Cut Problem for Fixed k." *Mathematics of Operations Research* 19 (1994), 24–37.

[GLS99] G. Gottlob, N. Leone and F. Scarcello. "Hypertree Decompositions and Tratable Queries." In *Proceedings of the 18th ACM Symposium on Database Systems* (1999), 21–32.

[Gr03] M. Grohe. "The Complexity of Homomorphism and Constraint Satisfaction Problems Seen from the Other Side." Manuscript, 2003.

[GSS01] G. Gottlob, F. Scarcello and M. Sideri. "Fixed Parameter Complexity in AI and Nonmonotonic Reasoning." To appear in *The Artificial Intelligence Journal*. Conference version in: *Proc. of the 5th International Conference on Logic Programming and Nonmonotonic Reasoning* (LPNMR'99), vol. 1730 of *Lecture Notes in Artificial Intelligence* (1999), 1–18.

[Hal03] M. Hallett. Presentation at the Annual Conference of the New Zealand Mathematics Research Institute, New Plymouth, January 2003.

[KM96] S. Khanna and R. Motwani. "Towards a Syntactic Characterization of PTAS." In: *Proc. STOC 1996*, ACM Press (1996), 329–337.

[KR00] S. Khot and V. Raman. "Parameterized Complexity of Finding Subgraphs with Hereditary properties." *Proceedings of the Sixth Annual International Computing and Combinatorics Conference (COCOON 2000)* July 2000, Sydney, Australia, Lecture Notes in Computer Science, Springer-Verlag 1858 (2000), 137-147.

[KS96] D.R. Karger and C. Stein. "A New Approach to the Minimum Cut Problem." *Journal of the ACM* 43 (1996), 601–640.

[Len83] H. W. Lenstra. "Integer Programming with a Fixed Number of Variables." *Mathematics of Operations Research* 8 (1983), 538–548.

[Luks82] E. Luks. "Isomorphism of Graphs of Bounded Valence Can Be Tested in Polynomial Time." *Journal of Computing and Systems Sciences 25* (1982), 42–65.

[MR99] M. Mahajan and V. Raman. "Parameterizing Above Guaranteed Values: MaxSat and MaxCut." *J. Algorithms* 31 (1999), 335-354.

[Nie02] R. Niedermeier. "Invitation to Fixed-Parameter Algorithms." Habilitationschrift, University of Tübingen, 2002.

[NR99] R. Niedermeier and P. Rossmanith. "Upper Bounds for Vertex Cover Further Improved." *Proceedings 16th STACS*, Springer-Verlag LNCS 1563 (1999), 561–570.

[PS03] E. Prieto and C. Sloper. "Either/Or: Using Vertex Cover Structure in Designing FPT Algorithms — the Case of k-Internal Spanning Tree." In this volume.

[RS95] N. Robertson and P. D. Seymour. "Graph Minors XIII. The Disjoint Paths Problem." *Journal of Combinatorial Theory, Series B* 63 (1995), 65–110.

[ST98] R. Shamir and D. Tzur. "The Maximum Subforest Problem: Approximation and Exact Algorithms." *Proc. ACM Symposium on Discrete Algorithms* (SODA'98), ACM Press (1998), 394–399.

[Wei98] K. Weihe, "Covering Trains by Stations, or the Power of Data Reduction," *Proc. ALEX'98* (1998), 1–8.