Some New Directions and Questions in Parameterized Complexity

Rodney G. Downey[1] and Catherine McCartin[2]

[1] Victoria University, Wellington, New Zealand
Rod.Downey@mcs.vuw.ac.nz
[2] Massey University, Palmerston North, New Zealand
C.M.McCartin@massey.ac.nz

Abstract. Recently there have been some new initiatives in the field of parameterized complexity. In this paper, we will report on some of these, concentrating on some open questions, and also looking at some of our ideas towards applying ideas of parametric complexity in the field of online model theory.

1 Introduction

Paremeterized complexity was developed as a tool to try address practical issues in algorithmic complexity. The basic idea is that the combinatorial explosion that occurs in exact algorithms for many intractable problems can systematically be address by seeking parameters to fix to contain this explosion. The main idea is that for *natural, practical* problems often there are one of more parameters that are naturally small, and *if it is these* that "make" the problem intractable then for a small range of these parameters the problem can be efficiently solved.

For example, if we are analyzing data arising as, for instance, the conflict graph of some problem in, say, computational biology. Because of the nature of the data we know that it is likely the conflicts are at most about 50 or so, but the data set is large, maybe 10^8 points. We wish to eliminate the conflicts, by identifying those 50 or fewer points. Let's examine the problem depending on whether the identification turns out to be a dominating set problem or a vertex cover problem. The classic two examples usually quoted here are the following.
Dominating Set. Essentially the only known algorithm for this problem is to try all possibilities. Since we are looking at subsets of size 50 or less then we will need to examine all $(10^8)^{50}$ many possibilities. Of course this is completely impossible.
Vertex Cover There is now an algorithm running in time $O(1.286^k + kn)$ ([19]) for determining if an G has a vertex c over of size k. This has been implemented and is practical for n of unlimited size and k up to around 400 [54].

The issue is *the manner by which the running time for a fixed k depends on the k. Critically, is k in the exponent of the size of the problem, or independent from that?* Consider the situation of a running time of $\Omega(n^k)$ vs $2^k n$.

There are myriads of natural implicit and explicit parameters by which problems can be FPT, including familiar tree-width metrics such as pathwidth, treewidth etc, as well as logical restrications.

To investigate the complexity of basic problems associated with persistence, we use the framework of *parameterized complexity theory*, introduced by Downey and Fellows [25]. We remind the reader that a parameterized language L is a subset of $\Sigma^* \times \Sigma^*$. If L is a parameterized language and $\langle \sigma, k \rangle \in L$ then we refer to σ as the *main part* and k as the *parameter*. The basic notion of tractability is *fixed parameter tractability* (FPT). Intuitively, we say that a parameterized problem is fixed-parameter tractable (FPT) if we can somehow confine the any "bad" complexity behaviour to some limited aspect of the problem, the parameter. Formally, we say that a parameterized language, L, is fixed-parameter tractable if there is a computable function f, an algorithm A, and a constant c such that for all k, $\langle x, k \rangle \in L$ iff $A(x,k) = 1$, and $A(x,k)$ runs in time $f(k)|x|^c$ (c is independent of k). For instance, k-Vertex Cover is solvable in time $\mathcal{O}(|x|)$. On the other hand, for k-Turing Machine Acceptance, the problem of deciding if a nondeterministic Turing machine with arbitrarily large fanout has a k-step accepting path, the only known algorithm is to try all possibilities, and this takes time $\Omega(|x|^k)$. This situation, akin to NP-completeness, is described by hardness classes, and reductions. A parameterized reduction, L to L', is a transformation which takes $\langle x, k \rangle$ to $\langle x', k' \rangle$, running in time $g(k)|x|^c$, with $k \mapsto k'$ a function purely of k.

Downey and Fellows [25] observed that these reductions gave rise to a hierarchy called the W-hierarchy.

$$FPT \subset W[1] \subseteq W[2] \subseteq \ldots \subseteq W[t] \subseteq \ldots.$$

The core problem for $W[1]$ is k-Turing Machine Acceptance, which is equivalent to the problem Weighted 3Sat. The input for Weighted 3Sat is a 3CNF formula, φ and the problem is to determine whether or not φ has a satisfying assignment of Hamming weight k. $W[2]$ has the same core problem except that φ is in CNF form, with no bound on the clause size. In general, $W[t]$ has as its core problem the weighted satisfiability problem for φ of the form "products of sums of products of ..." of depth t. It is conjectured that the W-hierarchy is proper, and from $W[1]$ onwards, all parametrically intractable.

There have been exciting new methods of establishing parametric tractability. In this paper we will not address them. Rather we point towards Rolf Niedermeier's survey [46], as well as Fellows survey [31], and Downey [23]. These is an upcoming book [47] which will be devoted to methods of parametric *tractability.*

There have been some new developments in the application of parameterized intractability to understand when PTAS's are likely feasible. These, and similar applications are highlight the use of parameterized complexity for exploring the boundary of feasibility *within polynomial time.* There have been very interesting developments exploring the running times of exact exponential algorithms, including the use "Mini-" classes of Downey et. al. [24]. These have been used especially by Fellows and his co-authors. This area is very promising and is the basis for an upcoming Dahstuhl meeting in 2005 as well as the *International Workshop in Parameterized and Exact Computation.* We will examine this material in Section2.

In Section 3, we examine the new classes generated by the notion of EPT from Flum, Grohe and Weyer [34]. In this paper, our first goal will be to examine these new notions and mention a number of natural open questions that they suggest. As solution to any of these problem would be significant and is an interesting case study in a more-or-less neglected arena: structural parameterized complexity. We will examine tese ideas in Section 3

The last goal of this article is to articulate a new program of the authors [26, 27]. devoted to applying the ideas of parameterized complexity, and topological graph theory to the online algorithms and online model theory. Our underlying idea is to provide a proper theoretical foundation to the study of online algorithms on online structures. Again as a case study we will look at online colourings of online graphs. Again we will highlight a number of open qwuestions in this area. This section is especially relevant to the present conference in view of the interest in *automatic* structures (such as Rubin's Thesis [52]). Here we have a structure, say, a graph, whose domain and operations are *presented* by automata, and there are natural exapmples of online structres since the domains and other aspects of thieir diagrams are given one point at a time, etc. More on this in Section 4.

2 $M[1]$, ETH and PTAS's

2.1 PTAS's

Let's re-examine the notion of P: classical *polynomial time.* Classical polynomial time allows for polynomials which can, in no way be regarded as "tractable". For instance, a running time of n^{9000} is certainly not feasible.

This fact has certainly been recognized since the dawn of complexity theory. The main argument used is that P is a robust class (in its closure properties) and "practical" problems in P have feasible running times. Certainly in the early 70's this point of view was correct, but recent general tools for extablishing times in P have give rise to bad running times.

When you are given some problem a classical approach is to either find a polynomial time algorithm to solve it, or to demonstrate that if is NP-hard. The latter would then suggest that no exact polynomial time algorithm exists.

The question is suppose that you have a problem that *is* in P, but the running time is hideous. What can you do? Parameterized complexity has been shown to be useful here. This is particularly true for bad running times in PTAS's (polynimla time approximation schemes). As per Garey and Johnson [45], polynomial time approximation schemes (PTAS's) are one of the main traditional methods for battling intractability. Many ingenious polynomial time approximation schemes have been invented for this reason. Often the wonderful PCP theorem of Arora *et al.* [5] shows that no such approximation exists (assuming

$P \neq NP$). But sometimes they do. Let's look at some recent examples, taken from some recent major conferences such as STOC, FOCS and SODA, etc. (See Downey [23], and Fellows [31] for more examples.)
• Arora [3] gave a $O(n^{\frac{3000}{\epsilon}})$ PTAS for Euclidean Tsp
• Chekuri and Khanna gave a $O(n^{12(\log(1/\epsilon)/\epsilon^8)})$ PTAS for Multiple Knapsack
• Shamir and Tsur [53] gave a $O(n^{2^{2^{\frac{1}{\epsilon}}}-1})$ PTAS for Maximum Subforest
• Chen and Miranda gave a $O(n^{(3mm!)^{\frac{m}{\epsilon}+1}})$ PTAS for General Multiprocessor Job Scheduling
• Erlebach *et al.* gave a $O(n^{\frac{4}{\pi}(\frac{1}{\epsilon^2}+1)^2(\frac{1}{\epsilon^2}+2)^2})$ PTAS for Maximum Independent Set for geometric graphs.

Table 1 below calculates some running times for these PTAS's with a 20% error.

Reference	Running Time for a 20% Error
Arora [3]	$O(n^{15000})$
Chekuri and Khanna [18]	$O(n^{9,375,000})$
Shamir and Tsur [53]	$O(n^{958,267,391})$
Chen and Miranda [21]	$> O(n^{10^{60}})$ (4 Processors)
Erlebach *et al.* [28]	$O(n^{523,804})$

Table 1. The Running Times for Some Recent PTAS's with 20% Error.

After the first author presented the table above at a recent conference (Downey [23]), one worker from the audience remarked to him "so that's why my code did not work," having tried to implement the Chen-Miranda algorithm!

Now sometimes the algorithms can be improved. For instance, Arora [4] also came up with another PTAS for Euclidean Tsp, but this time it was nearly linear and practical. The crucial question is : having gotten the algorithms and being unable to find a better algorithm, above how do we show that there are no paractical PTAS's. Remember we are *in P*, so lower bounds are hard to come by.

If the reader studies the examples above, they will realize that a source of the appalling running times is the $\frac{1}{\epsilon}$ in the exponent. We can define 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 cons tant. Then as was realized earlr in the game, setting $k = 1/\epsilon$ as the parameter, and then having a reduction to the PTAS from some parametrically hard problem will in essence demonstrate that no EPTAS exists..

Here is one early result in the program:

Theorem 1 (Bazgan [7], also Cai and Chen [15]). *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 one recent application of Bazgan's Theorem taken from Fellows, Cai, Juedes and Rosamond [16]. In a well-known paper, Khanna and Motwani introduced three planar logic problems towards an explanation of PTAS-approximability. Their suggestion is that "hidden planar structure" in the logic of an optim ization problem is what allows PTASs to be developed in Khanna and Motwani [39]. One of their core problems was the following.
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.

Theorem 2 (Fellows, Cai, Juedes and Rosamond [16]). Planar Tmin *is hard for $W[1]$ and therefore does not have an EPTAS unless $FPT = W[1]$.*

Fellows *et al.* [16] also show that the other two core problems of Khanna and Motwani [39] are also $W[1]$ hard and hence have no EPTAS's.

For most of the problems in Table 1 it is open which have EPTAS's. It is an open *project* to understand when problems such as those in [ACGKMP99], can have real EPTAS's rather than just PTAS's which have unrealistic running times. Recently Chen, Huang, Kanj, and Xia [20] have made significant progress by provding ea exact parametric classification for problems with *fully polynomial time approximation schemes*, for a wide class of problems (scalable problems):

Theorem 3 (Chen et. al. [20]). *Suppos ethat Q is a scalable NP optimization problem. Then Q has a FPTAS iff Q is in the class of PFTP of parameterized problems which can be solved by an algorithm whose running time is poolynomial in both $|x|$ and k.*

It seems reasonable that something of a similar ilk will be true for PTAS's. There are a number of similar applications using parameterized complexity for *practical* lowed bounds in polynomial time such as Alekhnovich and Razborov [2].

2.2 ETH

The previous section demonstrates that using parametric complexity we can address the classical issue of polynomial time approximation and whether there is an *efficient* polynomial time approximation. Recently the increased computational power available has shown that there are a lot of problems for which exponentail time algorithms can be useful in practice. This is particularly the case if the problem has an exponentail time algorithm which is significantly better than DTIME($2^{o(n)}$). However, there seems a "hard core" of problems for which not only do we think that there is no polynomial time algorithm for them, but in fact there is no subexponential time one either. This is the *exponential time hypothesis* first articulated in Impagliazzo ??.

• (ETH) n-variable 3-SATISFIABILITY is not solvable in DTIME($2^{o(n)}$).

Again it has been shwon that parametric techniques are extremely useful in showing that problems are likely not in DTIME($2^{o(n)}$). Actually, the idea that subexponentail time is intimately related to parametric complexity is relatively old, going back to Abrahamson, Downey and Fellows [1]. However, there has been a lot of interest in this area recently, especially after the realization of the central importance of the *Sparsification Lemma* of Impagliazzo, Paturi and Zane [37] by Cai and Juedes [17]. As we will see, showing that a problem is $W[1]$-hard would likely be enough (depending on the reduction). To address these issues, Downey, Estivill-Castro, Fellows, Prieto-Rodriguez and Rosamond [24] introduced a new class, the MINI-classes. Here the problem itself is parameterized. The notion is most easily explained by the following example.

MINI-3SAT
Input : A 3-CNF formula φ.
Parameter : k.
Problem If φ has size $\leq k \log n$ is it satisfiable?

The point here is the solution of the problem is akin to classical NP-completeness in that we are asking for an unrestricted solution to a restricted problem. We can similarly miniaturize any combinatorial problem. For instance, it is easy to see that MINI-CLIQUE is FPT. To remain in the miin-classes you need to make sure that the reductions are small. The core machine problem is a circuit problem: MINI-CIRCUIT SAT. This allows us to generate the class $M[1]$ of minaiturized problems. It is not hard to show that MINI-INDEPENDENT SET, MINI-VERTEX COVER, etc are all $M[1]$-complete. (Cai and Juedes [17], Downey et. al. [24]) It is unknown if MINI-TURING MACHINE ACCEPTANCE is $M[1]$ complete, as the reductions of the usual reductions are *not* linear, and we know of no such reduction. It is not hard to show that $FPT \subseteq M[1] \subseteq W[1]$. The relevance of this class to subexponential time is the following.

Theorem 4 (Cai and Juedes [17], Chor, Fellows and Juedes [32], Downey et. al. [24]). *The $M[1]$ complete problems such as* MIN-3SAT *are in FPT iff the exponential time hypothesis* fails.

For a recent survey summarizing this material with new proofs, we refer the reader to Flum and Grohe [33].

3 EPT and FPT

Now, let's re-examine the notion of *fixed parameter tractability.* Recall that $L \subset \Sigma^* \times \Sigma^*$ is FPT iff there is an algorithm deciding $\langle x, k\rangle \in L$ running in time $f(k)|x|^c$, with f an arbitary (computable) function, and c fixed, independent of k.

The criticisms of polynomial time can also be leveled, perhaps with even more force, at the notion of FPT since the function f can be arbitary. Remember, one of the claims of the theory is that it tries to address "practical" complexity. How does that claim stack up? One of the key methods of demonstrating *abstract* parametric tractability is the use of logical methods such as Courcelle's Theorem. Here one demonstrates that a given problem is for graphs of bounded treewidth and is definable in monadic second order logic, perhaps with counting. Then Courcelle's Theorem says the problem is linear time fixed-parameter tractable.

However, FPT is only really a general first approximation to feasability. The kinds of constants we get from applying Courcelle's Theorem are towers of 2's roughly of the order of the number of alternations of the monadic quantifiers. This was proven by Frick and Grohe [35]. These types of constants stand in contrast to constants gotten by elementary methods such as *bounded search trees*, and *crown reduction rules*, and *kernelization.* Here the parameter constants are managable, more like 2^k.

Flum, Grohe, anmd Weyer [34] recently introduced a new class to perhaps better address when a problem likely has a practical FPT algorithm.

Definition 1 (Flum, Grohe and Weyer [34]). *A parameterized problem is in EPT iff it is solvable in time $2^{0(k)}|x|^c$.*

For example, k-Vertex Cover is in EPT. Now the surprise. The reductions that Flum, Grohe and Weyer use are *not* parametric ones. The appropriate parametric reductions to keep within the class would be linear in k, and FPT in $|x|$. Flum, Frick and Grohe introduced their notion of an EPT reduction as being one that is *not parametric*, rather

$$L \leq_{EPT} L' \text{ iff } \langle x, k\rangle \in L \text{ iff } \langle x', k'\rangle \in L',$$

where $x \mapsto x'$ in time $2^{0(k)}|x|^c$, *but*

$$\langle k, x\rangle \mapsto k' \text{ has } k' \leq d(k + \log |x|).$$

This is, as the size of the problem grows, the slice of L' used for the reduction can slowly grow. Clearly, it is easy to prove that the two notions of reduction are distinct. Technically, the addition of the $\log |x|$ in the reduction allows for counters to be used in the reduction.

Frick, Flum and Grohe then go on to define a new hierarchy based upon these reduction notion called the E-hierarchy,

$$EPT \subseteq E[1] \subseteq E[2] \subseteq \dots,$$

which is robust for $t \geq 2$.

What is the point of all of this? Flum Grohe and Weyer demonstrate that a number of FPT problem lie at higher levels of this hierarchy and are therefore likely *not* EPT. For instance, classes of model-checking problems which Flum and Grohe showed did not have FPT algorithms with elementary parameter dependence are complete for various levels of the class. Another example of the phenomenom is the following.

Theorem 5 (Flum, Grohe and Weyer [34]). k-Vapnik-Chervonenkis Dimension *is complete for $E[3]$ under EPT reductions.*

We remark that k-Vapnik-Chervonenkis Dimension was known to be $W[1]$ complete under FPT reductions.

There are a number of very important FPT problems which have no known FPT algorithms which are also single exponential in the parameter. Showing that any of the following is likely not EPT would be very significant: k-Treewidth, k-Branchwidth, and k-Cutwidth. The same is true for many problems which also only have FPT algorithms known by Courcelle's Theorem or have been proven FPT (or PTIME) by treewdith methods.

Also, an interesting technical question is whether k-Independent Set is in $M[1]$ under EPT reductions. And what about randomization here.

We believe that the methods that are surely lacking here are suitable analogs of the PCP techniques.

4 Online Algorithms

In this section, we would like to discuss ideas from a new project of the authors which has several goals. They include (i) providing a theoretical foundation to *online model theory*, (ii) trying to apply methods from parameterized complexity in online algorithms, and (iii) seeking to understand the use of "promises" in this area.

The last 20 years has seen a revolution in the development of graph algorithms. This revolution has been driven by the systematic use of ideas from topological graph theory, with the use of graph width metrics emerging as a fundamental paradigm in such investigations. The role of graph width metrics, such as treewidth, pathwidth, and cliquewidth, is now seen as central in both algorithm design and the delineation of what is algorithmically possible. In turn, these advances cause us to focus upon the "shape" of much real life data. Indeed, for many real life situations, worst case, or even average case, analysis no longer seems appropriate, since the data is known to have a highly regular form, especially when considered from the parameterized point of view.

The authors have begun a project attempting to try to systematically apply the ideas from classical topological topological graph theory to online algorithms. In turn this has given rise to new parameters which would seem to have relavence in both the offline and online arenas.

Online problems come in a number of varieties. There seems no mathematical foundation to this area along the lines of finite model theory; and it is our intention to make such a theory. The basic idea is that an online problem is given one piece at a time per time step and our process must build the desired object according to this local knowledge. The following is a first definition for two such online structures.

Definition 2. *Let $\mathcal{A} = \langle A, R_1, \ldots, R_n \rangle$ be a structure. (We represent functions as relations for simplicity.)*

(i) *We will say that a collection $\mathcal{A}_s \subset \mathcal{A}_{s+1} \dots$ of finite substructures of $\mathcal{A}$ is a* monotone online presentation *where $\mathcal{A}_s$ denotes the restriction of $\mathcal{A}$ to the domain A_s.*
(ii) *More generally we can have $\mathcal{A} = \lim_s \mathcal{A}_s$, as above, but non-monotonically.*

For instance, an online graph could be one where we are given the graph one point at a time, and for each new vertex, I need to say which of the vertices seen so far are adjacent to the new vertex. Then non-montonic version, vertices may be added and then subtracted. Monotonic online structures model the situation where more and more information is given and we need to cope; for instance, bin packing; and the non-monotonic situation is more akin to a database that is changing with time.

An *online procedure* on an online presentation $\{\mathcal{A}_s : s \leq n\}$ is a computable function f, and a collection of structures $\{\mathcal{B}_s : \leq n\}$, such that

(i) $f : \mathcal{A}_s \mapsto \mathcal{B}_s$, and
(ii) $\mathcal{B}_s$ is an expansion of $\mathcal{A}_s$, and
(iii) f_{s+1} extends f_s.

The idea is that $\mathcal{B}$ would have some extra relation(s) which would need to be constructed in some online way. A nice example would be to consider colourings of some online graph. Here we would need to colour each new point witha new colour depending on only the information given by the finite graph coloured so far. The point is that the online situation is very different from the offline version. Every planar graph is 4-colourable, but there are trees of n vertices which have online presentations which need at least $\log n$ many colours for any online procedure to colour them. The *performance ratio* compares the offline and the online performances.

There are a constellation of questions here. For instance, suppose that $\mathcal{B}$ is the expansion of $\mathcal{A}$ by a single relation R, and this relation is first order definable. First order properties are essentially local. Hence one would expect that there is a general theorem which will show that there for any such first order R, there is an online procedure with reasonable performance ratio.

Then what about structures of bounded width? We could ask that the Gaifman graph of the structure have bounded (tree-)(branch-)(path-)width. There is almost nothing known here.

There is a wonderful analogy with computable structure theory, since we are dealing with strategies in games against a perhaps hostile universe. In some general sense, in a online situation it is very hard to use the finiteness of, say, a graph within the algorithm, since we don't know *how big* the graph might be. Information is only local. The recent work (at this conference!) on *automatic structures* where the algorithms in question deal with structures where the relations are given in an online was *by automata* are examples of this area. (See e.g. Rubin [52])

One area that has historically received some attention is online colouring of monotonic online graphs. Of course colouring is related to scheduling, and authors considered the situation where the pathwidth of the graph is bounded.

Kierstead and Trotter [41] gave an online algorithm for *interval graphs.* An interval graph $G = (V, E)$ is one for which there is a function g taking members of V to intervals in $\mathbb{R}$ wuch that if $v \neq v$, $g(v) \cap g(u) = \emptyset$. It is not hard to show that G has pathwidth k iff G is a subgraph of an interval graph with clique size bounded by $k + 1$. Clearly if I give you a presentation of an interval graph *as a pathwithd* $k + 1$ *graph*, then I can colour it in $k + 1$ colours, using first fit. But the topological fact that the graph has pathwidth $k+1$ will guarantee good persormance no matter how the graph is presented. Kierstead and Trotter [41] gave an online algorithm for interval graphs colouring them in $3k + 1$ many colours for any online presentation.

Kierstead and Qin showed that first-fit will online colour any k-inductive graph with at most $40(k+1)$ colours, and Chrobak and Slusarek [22] have shown that there is an online presentation of a pathwidth k graph for which first fit needs at least $4.4k$ many colours. It is open what the correct lower bound is here.

Related is Sany Irani's [38] notion of a *d-inductive* graph. A graph is d-inductive if there is an ordering $v_0, \ldots, v_n$ of its vertices such that for all i, v_i is adjacent to at most d vertices amongst $\{v_{i+1}, \ldots, v_n\}$.

For instance, a planar graph is 5-inductive. To see this, any planar graph must have a vertex of degree 5 or less. Call this v_0. Remove it, and edges adjacent to it. Repeat. Similarly, all graphs of treewidth k are k-inductive. This this notion generalizes the notions of bounded treewidth, bounded degree and planarity.

Theorem 6 (Irani [38]). *If G is d inductive, then first fit will colour any monotonic online presentation of G with at most $O(d \log |V|)$ many colours. This bound is tight.*

In [26], the authors showed that for trees of pathwidth k, first fit will do much better, needing giving $3k + 1$ many colours. It is unclear how the parameters of treeidth and pathwidth interact on number of colours needed.

We have also shown that Itani's bound is tight for bounded treewidth graphs.

Theorem 7. *For each k, there is a graph G of treewidth k and an online presentation of G for which first fit needs at least $k \log |V|$ many colours.*

Proof. To be put in.

There are still many unknowns here. For instance, what can be said on average? To make a *bad* online presentation of a graph of low pathwidth one seems to need to begin at the outer bags and work in. This would seema rare event. John Fouhy [36] ran some simulations and has found that in general for random pathwith k graphs we only ever seem to need $3k + 1$ colours for first-fit. This is not understood. Along with 0-1 behaviour, it is also suggestive of a more general theorem.

Finally this material gives ideas back to classical topological graph theory. The intuition behing a graph having low pathwidth is that it is "pathlike." In practice if we are thinking of some process which is pathlike in it graphical representation. then we would not think of it as a "fuzzy ball."

To make this idea precise, the authors introduced anew notion.

We say that a path decomposition of width k, in which every vertex of the underlying graph belongs to at most l nodes of the path, has pathwidth k and *persistence* l, and say that a graph that admits such a decomposition has *bounded persistence pathwidth*. We believe that this natural notion truly captures the intuition behind the notion of pathwidth.

A graph that can be presented in the form of a path decomposition with both low width and low persistence is properly pathlike, whereas graphs that have *high* persistence are, in some sense, "unnatural" or pathological. Consider the graph G presented in Figure 1. G is not really path-like, but still has a path decomposition of width only two. The reason for this is reflected in the presence of vertex a in *every* node of the path decomposition. Our underlying idea is that a pathwidth 2 graph should look more like a "long 2-path" than a "fuzzy ball".

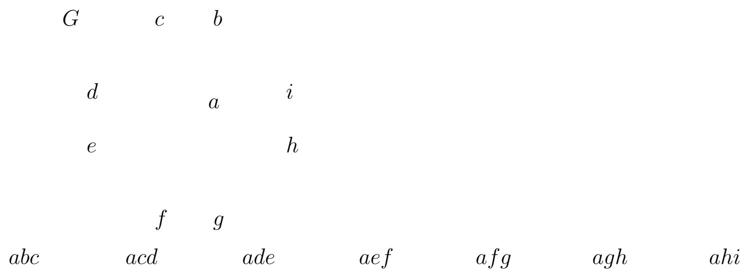

Fig. 1. A graph G having low pathwidth but high persistence.

What is intersting is that this notion is hard to recognize. Having persistence k is $W[2]$-hard to recognize (Downey and McCartin [26, 27]). Nevertheless, it seems that *if* we *know* that a graph has low persistence for a given pathwidth, we ought to be able to get better performance for algorithms. This is the very interesting situation where, because we know the shape of the input, we can explore algorithms which might be fast for the kinds of input we might expect, yet they should be slow in general. This idea remains to be explored.

References

1. K. Abrahamson, R. Downey and M. Fellows, *Fixed Parameter Tractability and Completeness IV: On Completeness for* $W[P]$ *and* PSPACE *Analogs, Annals of Pure and Applied Logic* 73 (1995), 235–276.

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

2. M. Alekhnovich and A. Razborov, *Resolution is Not Automatizable Unless W[P] is Tractable,* Proc. of the 42nd IEEE FOCS, 2001, 210-219.

3. 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, pp. 2–12.
4. S. Arora, "Nearly Linear Time Approximation Schemes for Euclidean TSP and Other Geomet ric Problems," *Proc. 38th Annual IEEE Symposium on the Foundations of Computing* (FOCS' 97), IEEE Press (1997), 554-563.
5. S. Arora, C. Lund, R. Motwani, M. Sudan and M. Szegedy, *Proof Verification and Intractability of Approximation Algorithms, Proceedings of the IEEE Symposium on the Foundations of Computer Science* (1992).
:
6. S. Arnborg, D. G. Corneil, and A. Proskurowski: *Complexity of finding embeddings in a k-tree.* SIAM J. Alg. Disc. Meth. 8, pp 277-284, 1987.
7. C. Bazgan, "Schémas d'approximation et complexité paramétrée," Rapport de stage de DEA d'Informatique à Orsay, 1995.
8. H. L. Bodlaender: *A linear time algorithm for finding tree decompositions of small treewidth.* SIAM J. Comput. 25, pp 1305-1317, 1996.
9. H. L. Bodlaender: *A partial k-arboretum of graphs with bounded treewidth.* Technical Report UU-CS-1996-02, Department of Computer Science, Utrecht University, Utrecht, 1996.
10. H. L. Bodlaender: *Treewdith: Algorithmic techniques and results.* Proc. 22nd MFCS, Springer-Verlag LNCS 1295, pp 19-36, 1997.
11. H. L. Bodlaender, J. Engelfreit: *Domino Treewidth.* J. Alg. 24, pp 94-127, 1997.
12. H. L. Bodlaender, M. R. Fellows, M. T. Hallett: *Beyond NP-completeness for problems of bounded width: Hardness for the W-hierarchy.* Proc. 26th Annual Symposium on Theory of Computing, pp 449-458, ACM Press, New York, 1994.
13. H. L. Bodlaender and T. Kloks: *Efficient and constructive algorithms for the pathwidth and treewdith of graphs.* J. Algorithms 21, pp 358-402, 1996.
14. H. L. Bodlaender, T. Kloks, and D. Kratsch: *Treewidth and pathwidth of permutation graphs.* Proceedings of the 20th International Colloquium on Automata, Langauges and Programming, A. Lingas, R. Karlsson, and S. Carlsson (Eds.), Vol 700 LNCS, Springer-Verlag, pp 114-125, 1993.
15. Liming Cai and J. Chen. "On Fixed-Parameter Tractability and Approximability of NP-Hard Optimization Problems," *J. Computer and Systems Sciences* 54 (1997), 465–474.
16. Liming Cai, M. Fellows, D. Juedes and F. Rosamond, *Efficient Polynomial-Time Approximation Schemes for Problems on Planar Stru ctures: Upper and Lower Bounds,* manuscript, 2001.
17. L. Cai and D. Juedes, *On the existence of subexponential time parameterized algorithms,* J. Comput. Sys. Sci., Vol. 67 (2003), 789-807. *The pathwidth and treewdith of cographs.* SIAM J. Disc. Meth. 6, pp 181-188, 1993.
18. 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. '
19. J. Chen, I.A. Kanj and W. Jia, *Vertex Cover: Further Observations and Further Improvements,* Journal of Algorithms, 41:280–301, 2001.
20. J. Chen, X. Huang, I. Kanj, and G. Xia, *Polynomial time approximation schemes and parameterized complexity, in Mathematical Foundations of Computer Science, 29th Annual Meeting, 2004, (J. Faila, V. Koubek, and J. Kratochvil, eds.) Springer-Verlag Lecture Notes in Computer Science, 3153, (2004), 500-512.*

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

22. M. Chrobak and M. Slusarek, On some packing problems related to dynamic storage allogation, *RARIO Inform. Theor. Appl., Vol. 22 (1988), 487-499.*

23. R. Downey, Parameterized complexity for the skeptic, *in Proceeding of the 18th Annual IEEE Conference on Comutational Complexity*, 2003.

24. 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 Rela ted Problems,* in *Proc. Australian Theory Symposium, CATS 2003*

25. R. G. Downey, M. R. Fellows: *Parameterized Complexity* Springer-Verlag, 1999.

26. R. G. Downey, C.M.McCartin: *Online Problems, Pathwidth, and Persistence* to appear in Proceedings of IWPEC 2004.

27. R. G. Downey, C.M.McCartin, *Bounded Persistence Pathwidth*, to appear.

28. T. Erlebach, K. Jansen and E. Seidel, *Polynomial Time Approximation Schemes for Geometric Graphs, Proc. ACM Symposium on Discrete Algorithms* (SODA'01), 2001, pp. 671–67 9.

29. B. de Fluiter: *Algorithms for Graphs of Small Treewidth.* ISBN 90-393-1528-0, 1997.

30. M. R. Fellows and M. A. Langston: *An analogue of the Myhill-Nerode theorem and its use in computing finite-basis characterizations.* Proceedings of the 30th Annual Symposium on Foundations of Computer Science, IEEE Computer Science Press, Los Alamitos, California, pp 520-525, 1989.

31. M. Fellows, *Parameterized complexity: the main ideas and connections to practical computing.* In: R. Fleischer et al. (Eds.) *Experimental Algorithmics,* LNCS 2547 (2002), 51–77.

32. M. Fellows, Personal communication, March 2003.

33. J. Flum and M. Grohe, *Parameterized complexity and subexponential time,* to appear Bulletin EATCS.

34. J. Flum, M. Grohe, and M. Weyer, *Bounded fixed-parameter tractability and* $\log^2$ *nondeterministic bits,* to appear Journal of Comput. Sys. Sci.

35. M. Frick and M. Grohe, *The Complexity of First-Order and Monadic Second-Order Logic Revisited,* in *LICS*, 2002, 215-224.

36. J. Fouhy, "Computational Experiments on Graph Width Metrics," MSc Thesis, Victoria University, Wellington, 2003.

37. R. Impagliazzo, R. Paturi and F. Zane, *Which problems have strongly exponential complexity?*, JCSS 63(4),: 512–530, 2001.

38. S. Irani *Coloring Inductive Graphs On-Line, Algorithmica*, vol.11, (1994), pp.53-72.

39. S. Khanna and R. Motwani, *Towards a Syntactic Characterization of PTAS,* in: *Proc. STOC 1996*, ACM Press (1996), 329–337.

40. H. Kierstead and J. Qin, *Colouring interval graphs with first-fit,* Discrete Math. Vol. 144 (1995), 47-57.

41. H. Kierstead and W. Trotter, *The linearity of first-fit colouring of interval graphs,* SIAM J. on Discrete Math., Vol. 1 (1988), 528-530.

42. J. Lagergren: *Efficient parallel algorithms for graphs of bounded treewidth.* J. Algorithms 20, pp 20-44, 1996.

43. J. Matousek and R. Thomas: *Algorithms for finding tree-decompositions of graphs.* J. Algorithms 12, pp 1-22, 1991.

44. C. McCartin: *Contributions to Parameterized Complexity* PhD Thesis, Victoria University, Wellington, 2003.

45. M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of* NP*-completeness.* W.H. Freeman, San Francisco, 1979.

46. R. Niedermeier, *Ubiquitous parameterization: invitation to fixed-parameter algorithms,* in Mathematical Foundations of Computer Science, 29th Annual Meeting, 2004, (J. Faila, V. Koubek, and J. Kratochvil, eds.) Springer-Verlag Lecture Notes in Computer Science, 3153, (2004), 84-103.
47. R. Niedermeier, *Fixed Parameter Algorithms,* Oxford University Press, in preparation.
48. L. Perkovic and B. Reed: *An Improved Algorithm for Finding Tree Decompositions of Small Width.* International Journal of Foundations of Computer Science 11 (3), pp 365-371, 2000.
49. B. Reed: *Finding approximate separators and computing treewdith quickly.* Proceedings of the 24th Annual Symposium on Theory of Computing, ACM Press, New York, pp 221-228, 1992.
50. N. Robertson and P. D. Seymour: *Graph minors - a survey.* Surveys in Combinatorics, I. Anderson (Ed.), Cambridge Univ. Press, pp 153-171, 1985.
51. N. Robertson, P. D. Seymour: *Graph minors II. Algorithmic aspects of tree-width.* Journal of Algorithms 7, pp 309-322, 1986.
52. S. Rubin, *Automatic Structures,* PhD Diss, Auckland University, 2004.
53. 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.
54. U. Stege, *Resolving Conflicts in Problems in Computational Biochemistry,* Ph.D. diss ertation, ETH, 2000.