Fixed-Parameter Tractability and Completeness I: Basic Results[1]

Rod G. Downey [2]
Mathematics Department
Victoria University
Wellington, New Zealand

Michael R. Fellows [3]
Computer Science Department
University of Victoria
Victoria, B.C. Canada

Abstract

For many fixed-parameter problems that are trivially solvable in polynomial-time, such as k-DOMINATING SET, essentially no better algorithm is presently known than the one which tries all possible solutions. Other problems, such as k-FEEDBACK VERTEX SET, exhibit *fixed-parameter tractability*: for each fixed k the problem is solvable in time bounded by a polynomial of degree c, where c is a constant independent of k. We establish the main results of a completeness program which addresses the apparent fixed-parameter intractability of many parameterized problems. In particular, we define a hierarchy of classes of parameterized problems $FPT \subseteq W[1] \subseteq W[2] \subseteq \cdots \subseteq W[SAT] \subseteq W[P]$ and identify natural complete problems for $W[t]$ for $t \geq 2$. (In other papers we have shown many problems complete for $W[1]$.) DOMINATING SET is shown to be complete for $W[2]$, and thus is not fixed-parameter tractable unless INDEPENDENT SET, CLIQUE, IRREDUNDANT SET and many other natural problems in $W[2]$ are also fixed-parameter tractable. We also give a compendium of currently known hardness results as an appendix.

1. Introduction

Many natural computational problems have input that consists of a pair of items. For

[1]Preliminary versions of some of the results of this paper were presented to the 21st Manitoba Conference on Numerical Mathematics and Computing, Winnipeg, Manitoba, Canada, October, 1991.These were reported in [DF1].

[2]Research supported by a grant from Victoria University IGC, and by the United States – New Zealand Cooperative Science Foundation Int 90-20558.

[3]Research supported by the United States Office of Naval Research under contract N00014-88K-0456, by the United States National Science Foundation under grant MIP-8919312, and by the National Science and Engineering Research Council of Canada.

example, the GRAPH GENUS problem is that of determining for an input pair (G, k), where G is a graph and k is a positive integer, whether the graph G embeds on the genus k surface. The problem of MINOR TESTING is that of determining for an input pair of graphs (G, H) whether H is a minor of G.

One of the reasons for our interest in parameterized problems, is that while many of these problems are NP-complete $PSPACE$-complete, or even provably intractable, it is sometimes the case that only a small range of parameter values are really important in practice, so that the (apparent) intractability of the general problem may be unduly pessimistic information. For many parameterized problems, we now have encouraging and perhaps useful fixed-parameter tractability results, such as the following.

Theorem 1.1 (Robertson and Seymour) [RS1] For every fixed graph H it can be determined in time $O(n^3)$ whether a graph G of order n has a minor isomorphic to H.

Theorem 1.2 (Bienstock and Monma) [BM] For every fixed k, it can be determined in time $O(n)$ whether a graph G of order n can be embedded in the plane so that k faces cover all the vertices.

Theorem 1.3 (Bodlaender) [Bod1] For every fixed k, it can be determined in time $O(n)$ whether a graph G of order n has a spanning tree with at least k leaves.

Theorem 1.4 (Lagergren) [La] For every fixed k, it can be determined in time $O(n \log^2 n)$ whether a graph G of order n has treewidth at most k.

Theorem 1.5 (Plehn and Voigt) [PV] For every fixed graph H of treewidth w, it can be determined in time $O(n^{w+1})$ whether a graph G of order n has a subgraph isomorphic to H. (Note that here the parameter is (a coding of) H)

Theorem 1.6 (Fellows and Langston) [FL1] For every fixed k, it can be determined in time $O(n)$ whether a graph G of order n has a cycle of length at least k.

Theorem 1.7 (Bodlaender, Downey and Fellows) [Bod2,DF1] For every fixed k, it can be determined in time $O(n)$ whether a graph G of order n contains k vertex-disjoint cycles.

For other parameterized problems, such as DOMINATING SET (Given a graph G and a positive integer k is there a set of k vertices in G such that every vertex either belongs to the set, or has a neighbour in the set) we have the contrasting situation where essentially no better algorithm is known than the "trivial" one which just exhaustively tries all possible solutions. For each fixed k, k- DOMINATING SET is solvable in this way in time $O(n^{k+1})$.

We make the following definitions in order to frame these complexity issues.

Definition. A *parameterized problem* is a set $L \subseteq \Sigma^* \times \Sigma^*$ where Σ is a fixed alphabet.

For a parameterized problem L and $y \in \Sigma^*$ we write L_y to denote the associated fixed-parameter problem (y is the parameter) $L_y = \{x|(x,y) \in L\}$. We refer to L_y as the y-th *slice* of L.

Definition. A parameterized problem L is (*weakly uniformly*) *fixed-parameter tractable* if there exists a constant α and an algorithm to determine if (x,y) is in L in time $f(|y|) \cdot |x|^\alpha$, where $f : N \to N$ is an arbitrary function. If f is recursive then we say that L is *strongly uniformly fixed-parameter tractable*. Finally, we say that L is nonuniformly f.p. tractable if there is a family of algorithms $\{\Psi_x : x \in N\}$ and a function f such that Ψ_x determines if (x,y) is in L in time $f(|y|) \cdot |x|^\alpha$.

In recent years a variety of methods useful for demonstrating fixed-parameter tractability have emerged, such as the well-quasiordering results of Robertson and Seymour [RS1,RS2], and general algorithmic methods for bounded treewidth [AF,Ar,ALS,BLW,Co,WHL].

The reader should note an important detail of the definition of fixed-parameter tractability given above. The results of Theorems 1.1–1.7 (and our Theorem 2.1 below) are clearly uniform; the proofs of these results can be implemented as a single algorithm that works for every parameter value. Consider, contrastingly, the consequence of Theorem 1 and the Graph Minor Theorem [RS2] that for each fixed k, it can be determined in time $O(n^3)$ whether a graph G of order n embeds on the surface of genus k. It is not immediately clear how these (infinitely many) distinct $O(n^3)$ algorithms, each based on a different finite obstruction set, can be combined into a single finite algorithm. This can done, however, by the two different methods of [FL1] and [FL2]. Almost all of the known fixed-parameter tractability results are (or can be made) uniform. While it is posible to construct examples ([DF3]) that show that the notions of tractability are indeed provably distinct, we also remark that there are natural examples of apparently all flavours of tractability. For instance consider the following examples.

(1.8) PLANAR IMPROVEMENT
Instance: A graph G.
Parameter: k.
Question: Is G a subgraph of a planar graph G' of diameter at most k?

(1.9) GRAPH LINKING NUMBER
Instance: A grapg G.
Parameter: k.
Question: Can G be embedded in 3-space so that at most k disjoint cycles are topologically linked?

We remark that (1.8) and (1.9) are both known to be f.p. tractable via the Robertson-Seymour theorems. But at present (1.8) is only known to be weakly uniformly tractable and (1.9) is only known to be nonuniformly tractable.

The difference between the known fixed-parameter complexity of DOMINATING SET and the problems addressed in Theorems 1.1–1.9, is analogous to the apparent complexity difference between NP-complete problems and problems in P. For most NP-complete problems we essentially know no better algorithm than the "trivial" one requiring exponential time which tries all possible solutions.

If $P = NP$ then DOMINATING SET is fixed-parameter tractable. A converse to this statement is not known, and is perhaps unlikely. The reasonable (we think) conjecture that DOMINATING SET is not fixed-parameter tractable is thus apparently stronger than the conjecture that $P \neq NP$. Certainly there is oracle evidence to perhaps support this claim. ([DF3]) The Graph Minor Theorem has the consequence that for each fixed surface we can decide graph embedability by employing *finitely many* minor tests. Thus the fixed-parameter tractability of MINOR TESTING leads to the fixed-parameter tractability of the GRAPH GENUS problem. This may be kept in mind as a motivating example for the following definition.

Definition. A (*uniform*) *reduction* of a parameterized problem L to a parameterized problem L' is an oracle algorithm A that on input (x, y) determines whether $x \in L_y$ and satisfies
(1) There is an arbitrary function $f : N \to N$ and a polynomial q such that the running time of A is bounded by $f(|y|)q(|x|)$.
(2) For each $y \in \Sigma^*$ there is a finite subset $J_y \subseteq \Sigma^*$ such that A consults oracles only for fixed-parameter decision problems L'_w where $w \in J_y$.

Of course in the above an oracle computation takes only one unit of time. If the oracle is consulted only once by A, then we will term the reduction *many:1*. As with the notion of tractability, there is a strong version. If the function f and the map taking y to J_y are both recursive, we say that the reduction is strongly uniform. (Similarly ther is a notion of nonuniform reduction, that we do not consider in detail here.) All of the results we prove in this paper hold for all of the frameworks, with the single exception of Theorem 4.1.

Lemma 1.1 If the parameterized problem L reduces to the parameterized problem L', and if L' is f.p. tractable, then L is f.p. tractable.

Proof. Let $f(|y|)q(|x|)$ be the bound on the running time of the reduction from L to L', and suppose L'_w is decidable in time $g(|w|) \cdot n^\alpha$. Wlog we can take f and g to be increasing. Let $y \in \Sigma^*$ and let $J_y \subseteq \Sigma^*$ be the associated finite subset of Σ^* for the reduction. Then we can determine if $(x, y) \in L$ in time $O(f(|y|)q(|x|)g(m)(f(|y|)q(|x|)^\alpha)$ where $m = \max\{|w| : w \in J_y\}$. □

Working Definition. Actually for the sake of most naturally occuring concrete reductions, we can take a simpler definition than the above. Most concrete reductions are m-reductions of the form $(x, k) \to (x', f(k))$ with x' depending upon x and k, running in time $h(k)|x|^\alpha$. The reason for this is that most natural problems are smooth in the sense that one has a natural

encoding of the slices with parameters below k into the k-th slice and hence we only need to look at one slice $f(k)$ for any input (x,k). The general definition is useful at times, and is certainly needed for structural theorems. It does the reader no harm to take the simplified definition for the remainder of the paper.

Remark: Alternative Definition. Another view of the ideas above is provided by Cai, Chen, Downey and Fellows [CCDF2], the *advice* view. In that paper Cai et al prove that if L is a parameterized language, then $L \in FPT$ iff there is a function $f : N \rightarrow \Sigma^*$, the advice function and a P-time oracle (deterministic) Turing Machine Φ, such that for all (x,k),

$$(x,k) \in L \text{ iff } \Phi^{f(k)}(x) \text{ accepts.}$$

That is if we allow for each k a finite piece of advice, then we can solve all the instances in time $|x|^\alpha$. There is similarly a relativized version for the reductions. We refer the reader to Cai et al [CCDF2] for more details.

The plan of this paper is as follows. In Section 2 we prove a particular combinatorial problem reduction that plays a key role in our main theorem, which is presented in Section 3. Section 4 summarizes our results and discusses open problems. During the long time that this paper has been under consideration many new hardness and completeness results have been found, and we see that this theory seems to have wide ranging consequences for the classification of parameterized hardness results. This is particularly true of, for instance, problems in molecular biology where one is often interested in a small parameter (e.g. the number of strands of DNA) yet the problem is very large (e.g. the length of the strand), and in VLSI design where we might have a small number of wafers of very large size. In the appendix we give a list of currently known hardness results, as well as a list of open classification problems.

To conclude this section we give some brief remarks concerning related investigations. As far as we are aware, the first person to suggest that something might be interesting in the fact that DOMINATING SET seems to require time $\Omega(n^{k+1})$ was Ken Regan[Re], in some comments in that paper. Regan did not pursue this issue. There have been investigations into "nondeterminism in P" such as the Kintala-Fischer β-hierarchy [KF] and the work of Buss and Goldsmith [BG] but these and similar related investigations were mainly structural, and looked at problems for a single k. One then cannot use P-time reductions since the class is in $DTIME(n^{f(k)})$ and usually $DTIME(n^k)$. These authors used instead small reductions such as *quasilinear* time reductions, which are of course machine dependent. The only paper to truely study the *asymptotic* parameterized behavior (i.e. the issue of n^α versus $n^{f(k)}$ with $f(k) \rightarrow \infty$) is Abrahamson, Ellis, Fellows and Mata [AEFM]. (Some of these results were recast in [BG].) In that paper the objects were P-checkable, P-indexed relations, called (polynomial time) *generator tester* pairs. For each k, one needed to be able to generate a P-time list of potential candidates for solutions that were easy to test. The actual definition is very involved but the following example gives the flavour. If we consider the problem VERTEX COVER, then for an instance G and a parameter k, the

potential witnesses would be pairs consisting of G and a vertex cover V with the index of $V \leq j = \binom{n}{0} + ... + \binom{n}{k}$. The ideas only seemed to apply to relations in NP. While there is a notion of parameterized tractability in [AEFM], it is roughly equivalent to our notion of nonuniform f.p. tractability and hence suffers from the problems that the tractable classes can be nonrecursive. The real problem with that paper is the notion of reducibility, which is defined on relations rather than parameterized languages, is rather unnatural and very unwieldy, and the notion of intractability is that of (essentially) being P-complete (or "dual P-complete") under *logspace* reducibility "by the slice." That is, in [AEFM], problems are PGT-complete (in their notation) only when for each k they are more or less P-complete. Of course this means that the [AEFM] ideas cannot address things such as DOMINATING SET and INDEPENDENT SET, nor apparently anything in the $W[t]$ classes. We remark that the [AEFM] results can be easily placed in our setup as they give $W[P]$-completeness results. (see, [ADF1,2]) We see our major contributions as being the identifying the correct notions of reducibility, identifying the correct setting for the study of parameterized intractibility, and finally identifying some "good" problems with which to measure hardness. The number of problems that have now been identified as $W[1]$ hard, and hence apparently intractable, would seem to support our claims. ($W[1]$ hardness is not analysed in the present paper, but in [DF2]. At this stage we will only remark that quite a number of parameterized problems have been shown to be $W[1]$-complete and we have a sort of Cook-Levin theorem for $W[1]$ which we feel strongly suggests its intractability. Namely Cai, Chen, Downey and Fellows [CCDF1] have shown that the following very generic problem is $W[1]$ complete.

SHORT TURING MACHINE COMPUTATION
Input: A nondeterministic Turing Machine M and a string x.
Parameter: k
Question: Does M have an accepting computation for x in k or fewer steps?)

To conclude, we finish by reiterating the remark that the notion of parameterized tractability is *not* a refinement of the classical notions arizing from NP-completeness, despite the fact that many of our examples arize from this arena. Problems can be $PSPACE$ complete (e.g. ALTERNATING HITTING SET, see [AEFM]) and yet have parameterized versions that are FPT. Also problems can be almost certainly not NP-complete unless some unlikely collapse occurs such as NP to $LOGNP$ and yet their parameterized versions can be $W[2]$ or $W[1]$ hard (e.g. the VAPNIK CHERVONENKIS DIMENSION, see [DEF]). Finally take any set A and consider the parameterized problem $\{(x, x) : x \in A\}$. Then this problem is just as hard as A classically so can even be provably intractable, and yet it is trivially FPT. These examples show that the parameterized complexity of problems and their unparameterized versions are pretty well unrelated, and thus our investigations point at a new dimension in the structure of problems.

2. A Key Combinatorial Reduction

Neither of the well-known computational problems of (1) determining whether a graph G has a dominating set of size k (DOMINATING SET), and (2) determining whether a graph G has an independent set of size k (INDEPENDENT SET) is know to be f.p. tractable, and it is perhaps a reasonable conjecture that they are not. The reader skeptical of this conjecture and willing to challenge it, will be advised by the results of this section to begin by working on INDEPENDENT SET, since a consequence of Theorem 2.1 is that Independent Set reduces to DOMINATING SET (and so the latter is "apparently harder" with respect to f.p. tractability). Presently the best known results for these problems are the trivial $O(n^{k+1})$ algorithm for DOMINATING SET, and a nontrivial algorithm for INDEPENDENT SET due to Nesetril and Poljak [NP], requiring time $O(n^{k(2+\epsilon)/3})$ where $2+\epsilon$ represents the best known exponent for fast matrix multiplication.

We show that WEIGHTED CNF SATISFIABILITY (defined below) reduces to DOMINATING SET. By the *weight* of a truth assignment to a set of boolean variables, we mean the number of variables assigned the value *true*, in the same way that the weight of a binary vector means the number of 1's in the vector. Since INDEPENDENT SET (and many other parameterized problems) easily reduce to this problem, we have the consequence claimed above. For example, a graph $G = (V, E)$ has a k-element independent set if and only if the expression $\prod_{uv \in E}(\bar{u} + \bar{v})$ has a weight k truth assignment. The notion of reduction that we use is (the working reduction) defined in Section 1.

WEIGHTED CNF SATISFIABILITY
Instance: A boolean expression X in conjuctive normal form. *Parameter:* k
Question: Is there a truth assignment of weight k that satisfies X?

Theorem 2.1 WEIGHTED CNF SATISFIABILITY strongly uniformly reduces to DOMINATING SET.

Proof. Let X be a Boolean expression in conjuctive normal form consisting of m clauses $C_1, ..., C_m$ over the set of n variables $x_0, ..., x_{n-1}$. We show how to produce in polynomial-time by local replacement, a graph $G = (V, E)$ that has a dominating set of size $2k$ if and only if X is satisfied by a truth assignment of weight k.

A diagram of the gadget used in the reduction is given in Diagram 1. The idea of the proof is as follows. There are k of the gadgets arranged in a circle. Each of the gadgets has 3 main parts. Taken clockwise from top to bottom, these are variable selection, gap selection and gap enforcement. The variable selection component is a clique and the gap selection consists of n cliques which we call columns. Our first action is to ensure that in *any* dominating set of $2k$ elements, we must pick one vertex from each of these two components. This goal is achieved by the $2k$ sets of $2k+1$ enforcers, vertices from V_4 and V_5. (The names refer to the sets below.) Take the V_4, for instance. For a fixed r, these $2k+1$ vertices are connected to all of the variable selection vertices in the component $A(r)$, and nowhere else. Thus if they are to be dominated by a $2k$ dominating set, then we must choose *some* element

in the set $A(r)$, and similarly we must choose an element in the set $B(r)$ by virtue of the V_5 enforcers. Since we will need exactly $2k$ (or even $\leq 2k$) dominating elements it follows that we must pick *exactly* one from each of the $A(r)$ and $B(r)$ for $r = 1, ..., k$.

As the name suggests these will be picked by the variable selection components, $A(r)$, $r = 0, ..., k-1$. Each of these k components consists of a clique of n vertices labelled $0, ..., n-1$. The intention being that the vertex labelled i represents a choice of variable i being made true in the formula X. Correspondingly in the next $B(r)$ we have columns (cliques) $i = 0, ..., n-1$. The intention is that column i corresponds to the choice of variable i in the preceding $A(r)$. The idea then is the following. We join the vertex $a[r, i]$ corresponding to variable i, in $A(r)$, to all vertices in $B(r)$ *except* those in column i. This means that the choice of i in $A(r)$ will cover all vertices of $B(r)$ except those in this column. It follows that we *must* choose the dominating element from this column and nowhere else. (There are no connections from column to column.) The columns are meant to be tha gap selection saying how many 0's there will be till the next positive choice for a variable. We finally need to ensure that if we chose variable i in $A(r)$ and gap j in column i from $B(r)$ then we need to pick $i + j + 1$ in $A(r + 1)$. This is the role of the gap enforcement component which consists of a set of n vertices (in V_6.) The method is to connect vertex j in column i of $B(r)$ to all of the n vertices $d[r, s]$ *except* to $d[r, i + j + 1]$. The point of this is that if we choose j in column i we will dominate all of the $d[r, s]$ except $d[r, i + j + 1]$. Since we will only connect $d[r, s]$ additionally to $a[r + 1, s]$ and nowhere else, to choose an element of $A[r + 1]$ and still dominate all of the $d[r, s]$ we must actually choose $a[r + 1, i + j + 1]$.

Thus the above provides a selection gadget that chooses k true variables with the gaps representing false ones. We enforce that the selection is consistent with the clauses of X via the clause variables V_3. These are connected in the obvious ways. One connects a choice in $A[r]$ or $B[r]$ corresponding to making a clause C_q true to the variable c_q. Then if we dominate all the clause variables too, we must have either chosen in some $A[r]$ a positive occurrence of a variable in C_q or we must have chosen in $B[r]$ a gap corresponding to a negative occorrence of a variable in C_q, and conversely. We now turn to the formal details.

The vertex set V of G is the union of the following sets of vertices:
$V_1 = \{a[r, s] : 0 \leq r \leq k-1, 0 \leq s \leq n-1\}$
$V_2 = \{b[r, s, t] : 0 \leq r \leq k-1, 0 \leq s \leq n-1, 1 \leq t \leq n-k+1\}$
$V_3 = \{c[j] : 1 \leq j \leq m\}$
$V_4 = \{a'[r, u] : 0 \leq r \leq k-1, 1 \leq u \leq 2k+1\}$
$V_5 = \{b'[r, u] : 0 \leq r \leq k-1, 1 \leq u \leq 2k+1\}$
$V_6 = \{d[r, s] : 0 \leq r \leq k-1, 0 \leq s \leq n-1\}$

For convenience, we introduce the following notation for important subsets of some of the vertex sets above. Let
$A(r) = \{a[r, s] : 0 \leq s \leq n-1\}$
$B(r) = \{b[r, s, t] : 0 \leq s \leq n-1, 1 \leq t \leq n-k+1\}$

$B(r,s) = \{b[r,s,t] : 1 \leq t \leq n-k+1\}$

The edge set E of G is the union of the following sets of edges. In these descriptions we implicitly quantify over all possible indices.
$E_1 = \{c[j]a[r,s] : x_s \in C_j\}$
$E_2 = \{a[r,s]a[r,s'] : s \neq s'\}$
$E_3 = \{b[r,s,t]b[r,s,t'] : t \neq t'\}$
$E_4 = \{a[r,s]b[r,s',t] : s \neq s'\}$
$E_5 = \{b[r,s,t]d[r,s'] : s' \neq s+t \pmod n\}$
$E_6 = \{a[r,s]a'[r,u]\}$
$E_7 = \{b[r,s,t]b'[r,u]\}$
$E_8 = \{c[j]b[r,s,t] : \exists i\ \overline{x_i} \in C_j, s < i < s+t\}$
$E_9 = \{d[r,s]a[r',s] : r' = r+1 \pmod n\}$

Suppose X has a satisfying truth assigment τ of weight k, with variables $x_{i_0}, x_{i_1}, ..., x_{i_{k-1}}$ assigned the value *true*. Suppose $i_0 < i_2 < ... < i_{k-1}$. Let $d_r = i_{r+1(\mathrm{mod}k)} - i_r \pmod n$ for $r = 0, ..., k-1$. It is straightforward to verify that the set of $2k$ vertices

$$D = \{a[r,i_r] : 0 \leq r \leq k-1\} \cup \{b[r,i_r,d_r] : 0 \leq r \leq k-1\}$$

is a dominating set in G.

Conversely, suppose D is a dominating set of $2k$ vertices in G. The closed neighborhoods of the $2k$ vertices $a'[0,1], ..., a'[k-1,1], b'[0,1], ..., b'[k-1,1]$ are disjoint, so D must consist of exactly $2k$ vertices, one in each of these closed neighborhoods. Also, none of the vertices of $V_4 \cup V_5$ are in D, since if $a'[r,u] \in D$ then necessarily $a'[r,u'] \in D$ for $1 < u' < 2k+1$ (otherwise D fails to be dominating), which contradicts that D contains exactly $2k$ vertices. It follows that D contains exactly one vertex from each of the sets $A(r)$ and $B(r)$ for $0 \leq r \leq k-1$.

The possibilities for D are further constrained by the edges of E_4, E_5 and E_9. The vertices of D in V_1 represent the variables set to *true* in a satisfying truth assignment for X, and the vertices of D in V_2 represent intervals of variables set to *false*. Since there are k variables to be set to *true* there are, considering the indices of the variables mod n, also k intervals of variables to be set to *false*.

The edges of E_4, E_5 and E_9 enforce that the $2k$ vertices in D must represent such a choice consistently. To see how this enforcement works, suppose $a[3,4] \in D$. This represents that the third of k distinct choices of variables to be given the value *true* is the variable x_4. The edges of E_4 force the unique vertex of D in the set $B(3)$ to belong to the subset $B(3,4)$. The index of the vertex of D in the subset $B(3,4)$ represents the difference (mod n) between the indices of the third and fourth choices of a variable to receive the value *true*, and thus the vertex represents a range of variables to receive the value *false*. The edges of E_5 and E_9 enforce that the index t of the vertex of D in the subset $B(3,4)$ represents the "distance" to the next variable to be set *true*, as it is represented by the unique vertex of D in the set

$A(4)$.

It remains only to check that the fact that D is a dominating set insures that the truth assigment represented by D satisfies X. This follows by the definition of the edge sets E_1 and E_8. □

Because DOMINATING SET can be easily reduced to WEIGHTED CNF SATISFIABILITY with no negated literals, the above theorem shows the surprising fact that WEIGHTED SATISFIABILITY reduces to MONOTONE WEIGHTED CNF SATISFIABILITY. Interpreted in terms of circuits, this combinatorial reduction plays a crucial role in the fundamental completeness results surveyed in the next section.This reduction also allows for a number of other applications. For instance, consider the problem below.

WEIGHTED $\{0,1\}$-INTEGER PROGRAMMING
Instance: A binary matrix A and a binary vector $\mathbf{b}$.
Parameter: k
Question: Does $A \cdot \mathbf{x} \geq \mathbf{b}$ have a binary solution of weight k?

We have the following:

Corollary WEIGHTED CNF SATISFIABILITY reduces to WEIGHTED $\{0,1\}$- INTEGER PROGRAMMING

Proof Let (X,k) be an instance of MONOTONE WEIGHTED CNF SAT. Let $C_1,...,C_p$ list the clauses of X and $x_1,...,x_m$ list the variables. Let A be the matrix $\{a_{i,j} : i = 1,...p, j = 1,...,m\}$ with $a_{i,j} = 1$ if x_j is present in C_i and $a_{i,j} = 0$ otherwise. Let $\mathbf{b}$ be the vector with 1 in the j-th position for $j = 1,...,p$. It is easy to see that $A \cdot \mathbf{x} \geq \mathbf{b}$ has a solution of weight k iff X has a satisfying assignment of weight k (and the reasoning is reversible). □

3. A Completeness Theory for Fixed-Parameter Intractability

In order to frame a completeness theory to address the apparent fixed-parameter intractability of DOMINATING SET and other problems, we need to define appropriate classes of parameterized problems. As one perhaps might expect satisfiability occupies a central role in our investigations. But now the situation is apparently much more complex than in the classical case. For instance, while classically there is no difference between CNF SATISFIABILITY and SATISFIABILITY for general formulae in propositional logic, there seems to be *no* parameterized reduction computing general SATISFIABILITY from CNF SATISFIABILITY and hence they do seem to be genuinely of different complexity from a parameterized point of view. Considerations such as these lead to the conclusion that there seem to be *many* different parameterized degree classes of natural problems. This is quite different from the situation in classical, say, NP-completeness results where virtually all natural problems which are not in P seem to be NP complete.

Current Measure of Intractability: At the present time if we wish to prove parameterized intractability, we show that the problem at hand can compute WEIGHTED 3CNF SATISFIABILITY. This is the class we call $W[1]$ and as we said before, we seem to have very strong evidence that it is intractable. We refer the reader to [DF2].

t-Normalized Formulae Now in fact, there does not seem to be any reduction from parameterized CNF to 3CNF. This leads naturally to the realization that the logical depth of a propositional formula affects its parameterized complexity. Thus if CNF is thought of as products-of-sums of literals, then we can define a propositional formula to be *t-normalized* if it is of the form products-of-sums-of-products... of literals with t-alternations. (So hence 2-normalized is the same as CNF.) Now we believe that for all t, WEIGHTED SATISFIABILITY for t-normalized formulae is strictly easier than WEIGHTED SATISFIABILITY for $t+1$-normalized formulae. This suggests a hierarchy based on the complexity of t-normalized formulae. It turns out that this hierarchy is quite useful for the classification of the complexity of many natural problems, and t-normalized formulae do give quite a bit more computational power. We make this all more precise through the introduction of the circuit-based classes below.

The classes that we define below are intuitively based on the complexity of the circuits required to check a solution. The size of a circuit is as usual the number of gates in the circuit.

We first define decision circuits in which some gates have bounded fan-in and some have unrestricted fan-in. It is assumed that fan-out is never restricted.

Definition. A Boolean circuit is of *mixed type* if it consists of circuits having gates of the following kinds.

(1) *Small gates*: *not* gates, *and* gates and *or* gates with bounded fan-in. We will usually assume that the bound on fan-in is 2 for *and* gates and *or* gates, and 1 for *not* gates.

(3) *Large gates*: *And* gates and *Or* gates with unrestricted fan-in.

We will use lower case to denote small gates (*or* gates and *and* gates), and upper case to denote large gates (*Or* gates and *And* gates).

Definition. The *depth* of a circuit C is defined to be the maximum number of gates (small or large) on an input-output path in C. The *weft* of a circuit C is the maximum number of large gates on an input-output path in C.

Definition. We say that a family of circuits F has *bounded depth* if there is a constant h such that every circuit in the family F has depth at most h. We say that F has *bounded weft* if

there is constant t such that every circuit in the family F has weft at most t. F is *monotone* if the circuits of F do not have not-gates. A circuit C is a *decision circuit* if it has a single output. A decision circuit C *accepts* an input vector x if the single output gate has value 1 on input x.

Definition. Let F be a family of decision circuits. We allow that F may have many different circuits with a given number of inputs. To F we associate the parameterized circuit problem $L_F = \{(C,k) : C \in F \text{ accepts an input vector of weight } k\}$.

Definition. A parameterized problem L belongs to $W[t]$ (*monotone* $W[t]$) if L uniformly reduces to the parameterized circuit problem L_F for the family $F_{h,t}$ of depth h, mixed type (monotone) decision circuits of weft at most t.

Definition. We denote the class of fixed-parameter tractable problems FPT.

Thus we have the containments

$$FPT \subseteq W[1] \subseteq W[2] \subseteq ...$$

and we conjecture that each of these containments is proper. We term the union of these classes together with two other classes $W[SAT] \subseteq W[P]$, the W Hierarchy. Here $W[P]$ denotes the class obtained by having no restriction on depth, i.e. P-size circuits, and $W[SAT]$ denotes the restriction to boolean circuits of P-size. We do not explore $W[SAT]$ nor $W[P]$ here but do so in [ADF1,2].

Our main result shows that WEIGHTED CNF SATISFIABILITY is complete for $W[2]$ and that similar problems are complete for each level of the W Hierarchy of parameterized problem classes. This theorem, Theorem 3.1, plays a role in our theory analogous to Cook's theorem for NP-completeness, in the following sense. Usually membership of a particular $W[t]$ class is easy so the circuit definition is easy to reduce *to*, while the t-normailzed formulae provide problems that are easy to reduce *from* to establish hardness. The other view of the Cook(-Levin) theorem is that it connects a generic problem from Turing Machines to SATISFIABILITY and hence SATISFIABILITY is very unlikely to be tractable. This other view of the Cook(-Levin) theorem is not pursued here but is established for $W[1]$ in [DF2], as we mentioned earlier. The present paper, and particularly Theorem 3.1 provide an important technical framework for hardness results.

It is interesting that the combinatorial reduction of Theorem 2.1 plays a key role (as a "change of variables") in our proof of Theorem 3.1. Thus the entire argument that DOMINATING SET is complete for $W[2]$ actually uses this combinatorial reduction *twice*. Recall the notion of t-normalized formula we discussed earlier. It naturally gives rise to the following problem.

WEIGHTED t-NORMALIZED SATISFIABILITY
Input: A t-normalized boolean expression X.

Parameter: A positive integer k.
Question: Does X have a satisfying truth assignment of weight k?

Theorem 3.1 For $t \geq 2$ WEIGHTED t-NORMALIZED SATISFIABILITY is many:1 complete for $W[t]$.

Proof. Let $L \in W[t]$. Let F be the family of circuits of depth bounded by h and weft bounded by t to which L reduces. It suffices to reduce L_F to WEIGHTED t-NORMALIZED SATISFIABILITY. An instance of the latter problem may be viewed as a pair consisting of a positive integer k and a circuit having t alternating layers of *And* and *Or* gates corresponding to the t-normalized expression structure P-o-S-o-P-..., and having a single output *And* gate. Thus the argument essentially shows how to "normalize" the circuits in F.

Let (C, k) be an instance of L_F. We show how to determine whether C accepts a weight k input vector, by consulting an oracle for WEIGHTED t-NORMALIZED SATISFIABILITY (viewed as a problem about circuits) for finitely many weights k'. The algorithm for this determination will be uniform in k, and run in time $f(k)n^{\alpha}$ where n is the size of the circuit C. The exponent α will be a (possibly exponential) function of h and t. This is permissible, since every circuit in F observes these bounds on depth and weft.

Step 1. The reduction to tree circuits.

The first step is to transform C into a *tree circuit* C' (or *formula*) of depth and weft bounded by h and t, respectively. In a tree circuit every logic gate has fan-out one. (The input nodes may have large fan-out.) The transformation is accomplished by replicating the portion of the circuit above a gate as many times as the fan-out of the gate, beginning with the top level of logic gates, and proceeding downward level by level. (We regard a decision circuit as arranged with the inputs on top and the output on the bottom.) The creation of C' from C may require time $O(n^{O(h)})$ and involve a similar blow-up in the size of the circuit. The tree circuit C' accepts a weight k input vector if and only if the original circuit C accepts a weight k input vector.

Step 2. Moving the *not* gates to the top of the circuit.

Let C denote the circuit we receive from the previous step (we will use this notational convention throughout the proof). Transform C into an equivalent circuit C' by commuting the *not* gates to the top, using DeMorgan's laws. This may increase the size of the circuit by at most a constant factor. The tree circuit C' thus consists (from the top) of the input nodes, with *not* gates on some of the lines fanning out from the inputs. In counting levels we consider all of this as level 0, and may refer to negated fan-out lines from the input nodes as negated inputs. Next, there are levels consisting only of large and small *and* and *or* gates, with a single output gate (which may be of either principal logical denomination at this point).

Step 3. Homogenizing the layers.

The goal of this step is to reduce to the situation where all of the large gates are at the bottom of the circuit, in alternating layers of large *And* and *Or* gates. To achieve this we work from the bottom up, with the first task being to arrange for the output gate to be large.

Let C denote the circuit received from the previous step. Suppose the output gate z is small. Let $C[z]$ denote the connected component of C including z that is induced by the set of small gates. Thus all gates providing input to $C[z]$ are either large or are input gates of C. Because of the bound h on the depth of C, there are at most 2^h inputs to $C[z]$. The function of these inputs computed by $C[z]$ is equivalent to a product-of-sums expression E_z having at most 2^{2^h} sums, with each sum a product of at most 2^h inputs. Let C' denote the circuit equivalent to C obtained by replacing the small gate output component $C[z]$ with E_z, duplicating subcircuits of C as necessary to provide the inputs to the depth 2 circuit representing E_z. (The "product" gate of E_z is now the output gate of C'.) This entails a blowup in size by a factor bounded by 2^{2^h}. Since h is an absolutely fixed constant (not dependent on n or k) this blowup is "linear" and permitted. Note that E_z and therefore C' are easily computed in a similar amount of time to this size blowup.

Let p denote the output *and* gate of C' (corresponding to the product in E_z). Let $s_1, ..., s_m$ denote the *or* gates of C' corresponding to the sums of E_z. We consider all of these gates to be *small*, since the number of inputs to them does not depend on n or k. (Equivalently, if the gates of these two levels were replaced by binary input gates, we would see that the reduction of C to C' has increased circuit depth from h to 2^h.)

Each *or* gate s_i of C' has 3 kinds of input lines: those coming from large *And* gates, those coming from large *Or* gates, and those coming from input gates of C'. We will use the same symbol to denote an input line, the subcircuit of C' that computes the value on that line, or the Boolean expression corresponding to the subcircuit (since C' is a tree circuit, it is equivalent to a Boolean expression). Let these 3 groups of inputs be denoted, respectively:

$$S_{i,\wedge} = \{s_i[\wedge, j] : \quad j = 1, ..., m_{i,\wedge}\}$$

$$S_{i,\vee} = \{s_i[\vee, j] : \quad j = 1, ..., m_{i,\vee}\}$$

$$S_{i,\top} = \{s_i[\top, j] : \quad j = 1, ..., m_{i,\top}\}$$

and define

$$S_i = S_{i,\wedge} \cup S_{i,\vee} \cup S_{i,\top}$$

For each line $s_i[\vee, j]$ of C' coming from a large *Or* gate u, let

$$S_{i,\vee,j} = \{s_i[\vee, j, k] : \ k = 1, ..., m_{i,\vee,j}\}$$

denote the set of input lines to u in C'. Similarly, for each line $s_i[\wedge, j]$ of C' coming from a large And gate v, let

$$S_{i,\wedge,j} = \{s_i[\wedge, j, k] : \ k = 1, ..., m_{i,\wedge,j}\}$$

denote the set of input lines to v in C'.

Let

$$k' = \sum_{i=1}^{m} (1 + m_{i,\vee})$$

The integer k' is the number of or gates (counting both large and small gates) that are either part of $C[z]$ or directly supply input to $C[z]$. Note that k' is bounded above by $2^h \cdot 2^{2^h}$.

We describe how to produce a weft t circuit C'' from C' that accepts an input vector of weight $k'' = k + k'$ if and only if C' (and therefore C) accepts an input vector of weight k. The tree circuit C'' will have a large And gate giving the output.

Let $x_1, ..., x_n$ denote the inputs to C'. The circuit C'' has additional input variables that, for the most part, correspond to the input lines to the or gates singled out for attention above. The set V of new input variables is the union of the following groups of variables:

$$V = \left(\bigcup_{i=1}^{m} V_i \right) \cup \left(\bigcup_{i=1}^{m} \bigcup_{j=1}^{m_{i,\vee}} V_{i,j} \right)$$

where

$$V_i = \{v_i[\wedge, j] : \ 1 \leq j \leq m_{i,\wedge}\} \cup \{v_i[\vee, j] : \ 1 \leq j \leq m_{i,\vee}\} \cup \{v_i[\top, j] : \ 1 \leq j \leq m_{i,\top}\}$$

and

$$V_{i,j} = \{v_i[\vee, j, k] : \ 1 \leq k \leq m_{i,\vee,j}\} \cup \{n[i, j]\}$$

The circuit C'' is represented by the Boolean expression:

$$C'' = E_1 \cdot E_2 \cdot E_3 \cdot E_4 \cdot E_5 \cdot E_6 \cdot E_7$$

where

$$E_1 = \prod_{i=1}^{m} \left(\sum_{u \in V_i} u \right)$$

$$E_2 = \prod_{i=1}^{m} \prod_{u \neq v, \ u,v \in V_i} (\neg u + \neg v)$$

$$E_3 = \prod_{i=1}^{m} \prod_{j=1}^{m_{i,\vee}} \left(\sum_{u \in V_{i,j}} u \right)$$

$$E_4 = \prod_{i=1}^{m} \prod_{j=1}^{m_{i,\vee}} \prod_{u \neq v}^{u,v \in V_{i,j}} (\neg u + \neg v)$$

$$E_5 = \prod_{i=1}^{m} \prod_{j=1}^{m_{i,\wedge}} \prod_{k=1}^{m_{i,\wedge,j}} (s_i[\wedge, j, k] + \neg v_i[\wedge, j])$$

$$E_6 = \prod_{i=1}^{m} \prod_{j=1}^{m_{i,\vee}} (\neg v_i[\vee, j] + \neg n[i, j])$$

$$E_7 = \prod_{i=1}^{m} \prod_{j=1}^{m_{i,\vee}} \prod_{k=1}^{m_{i,\vee,j}} (s_i[\vee, j, k] + \neg v_i[\vee, j, k])$$

The size of C'' is bounded by $|C'|^2$.

Claim 1. The circuit C'' has weft t.

To see this, note first of all that since $t \geq 2$, any input-output path beginning from a new input variable (in V) has at most 2 large gates as the expression for C'' is essentially a product-of-sums. In E_5 and E_7 the sums involve subexpressions of C'; any input-output path from an original input variable (of C') passes through one of these *or* gates. Observe that in C' these subexpressions have weft at most $t-1$. The sums of E_5 and E_7 are small, so these paths further encounter only the bottommost large *And* gate.

Claim 2. The circuit C'' accepts an input vector of weight k'' if and only if C' accepts an input vector of weight k.

First note that any input vector of weight k'' accepted by C'' must set exactly one variable in each of the sets of variables V_i (for $i = 1, ..., m$) and $V_{i,j}$ (for $i = 1, ..., m$ and $j = 1, ..., m_{i,\vee}$) to 1 and all of the others in the set to 0 in order to satisfy $E_1 \cdot E_2 \cdot E_3 \cdot E_4$. It follows that any such accepted input must set exactly k of the original variables of C' to 1, by the definition of k''.

The role of the (new) variables set to 1 in the sets of variables that represent inputs to *or* gates is to indicate an accepting computation of C' on the weight k input of old variables. The expressions E_5,...,E_7 enforce the correctness of this representation in C'' of the computation of C'.

The expression E_5 insures that if the new variable $v_i[\wedge, j]$ is set to 1, indicating that the subexpression $s_i[\wedge, j]$ of C' evaluates to 1, then every argument $s_i[\wedge, j, k]$ must evaluate to 1. (Note that subexpressions $s_i[\wedge, j, k]$ appear in C'' while the subexpressions $s_i[\wedge, j]$ do not. The computations performed in C' by the latter are simply represented by the values of the input variables in V.) The role of the variables $n[i, j]$ is to represent that "none of the inputs" to the *or* gate has the value 1. The expression E_6 enforces that if this situation

is represented, then the output of the gate is not represented as having the value 1. The expression E_7 insures that if the new variable $v_i[\vee, j, k]$ has the value 1, indicating that the subexpression $s_i[\vee, j, k]$ of C' evaluates to 1, then this subexpression must in fact evaluate to 1.

By the above, we may now assume that the circuit we are working with has a large output gate (which may be of either denomination). Renaming for convenience, let C denote the circuit we are working with under this assumption.

If g and g' are gates in C of the same logical character ($\wedge$ or $\vee$) with the output of g going to g', then they can be consolidated into a single gate without increasing weft if g is small and g' is large. We term this a *permitted contraction.* Note that if g is large and g' is small then contraction may not preserve weft. We will assume that permitted contractions are performed whenever possible, interleaved with the following two operations.

(1) *Replacement of bottommost small gate components.*

As at the beginning of Step 3, let $C_1, ..., C_m$ denote the bottommost connected components of C induced by the set of small gates, and having at least one large gate input. Since the output gate of C is large, each C_i gives output to a large gate g_i. If g_i is an *And* gate, then C_i should be replaced with product-of-sums circuitry equivalent to C_i. If g_i is an *Or* gate, then C_i should be replaced with equivalent sum-of-products circuitry. Note that in either case this immediately creates the opportunity for a permitted contraction. As per the discussion at the beginning of Step 3, this replacement circuitry is small, and this operation may increase the size of the circuit by a factor of 2^{2^h}. This step will be repeated at most h times, as we are working from the bottom up in transforming C.

(2) *Commuting small gates upwards.*

After (1), and after the permitted contractions, the bottommost small gate components are each represented in the modified circuit C' by a single small gate h_i giving output to g_i. Without loss of generality, all of the arguments to h_i may be considered to come from large gates. (The only other possibility is that an input argument to h_i may be from the input level of the circuit – but there is no increase in weft in treating this as an honorary large gate for convenience.) Suppose that g_i is an *And* gate, and that h_i is an *or* gate (the other possibility is handled dually).

There are three possible cases:
(i) All of the arguments to h_i are from *Or* gates.
(ii) All of the arguments to h_i are from *And* gates.
(iii) The arguments to h_i include both large *Or* gates and large *And* gates.

In case (i), we may consolidate h_i and all of the gates giving input to h_i into a single large *Or* gate without increasing weft.

In case (ii), we replace the small $\vee$ (h_i) of large $\wedge$'s with the equivalent (by distribution) large $\wedge$ of small $\vee$'s. Since h_i may have 2^h inputs, this may entail a blowup in the size of the circuit from n to n^{2^h}. This does not increase weft, and creates the opportunity for a permitted contraction.

In case (iii), we similarly replace h_i and its argument gates with circuitry representing a product-of-sums of the inputs to the arguments of h_i. The difference is that in this case, the replacement is a large $\wedge$ of large (rather than small) $\vee$ gates. Weft is preserved when we take advantage of the contraction now permitted between the large $\wedge$ gate and g_i.

We may achieve our purpose in this step by repeating the cycle of (1) and (2). At most h repetitions are required. The total blowup in the size of the circuit in this step is crudely bounded by $n^{2^{h^2}}$.

Step 4. Removing a bottommost *Or* gate.

By a Turing reduction, we can determine whether a tree circuit giving output from an *Or* gate accepts a weight k input vector, by simply making the same determination for each of the input branches (subformulae) to the gate.

In order to accomplish this step by a many:1 reduction, we do the following. Let b be the number of branches of the circuit C with bottommost *Or* gate that we receive at the beginning of this step. We modify C by creating new inputs $x[1], \ldots, x[b]$. The purpose of these input variables is to indicate which branch of C accepts a weight k input vector. Let $C_1, \ldots, C_b$ be the branches of C, so that C is represented by the expression $C_1 + \cdots + C_b$. The modified circuit C' is represented by the expression

$$C' = (x[1] + \cdots + x[b]) \cdot \prod_{1 \leq i < j \leq b} (\neg x[i] + \neg x[j]) \cdot \prod_{1 \leq i \leq b} (C_i + \neg x[i])$$

The first two product terms of the above expression insure that exactly one of the new variables must have value 1 in an accepted input vector. The modified circuit C' accepts a weight $k+1$ input vector if and only if C accepts a weight k input vector. For weft at least two, the transformation is weft-preserving, and yields a circuit C' with bottommost *And* gate, but possibly with *not* gates at the lower levels. Thus it may be necessary to repeat steps 2 and 3 to obtain a homogenized circuit with bottommost *And* gate.

Step 5. Organizing the small gates.

The tree circuit C received from the previous step has the properties: (i) the output gate is an *And* gate, (ii) from the bottom, the circuit consists of layers which alternately consist of only *And* gates or only *Or* gates, for up to t layers, and (iii) above this, there are branches B of height $h' = h - t$ consisting only of small gates. Since a small gate branch B has bounded depth, it has at most $2^{h'}$ gates, and thus in constant time (since h is fixed), we

can find either (1) an equivalent sum-of-products circuit with which to replace B, as required by Case 1 of step 6 below, or (2) an equivalent product-of-sums circuit, as required by Case 2 of step 6.

In this step, all such small gate branches B of C are replaced in this way, appropriately for the relevant case of step 6. In Case 1, the depth 2 sum-of-products circuits replacing the small gate branches B have a bottomost *or* gate g_B of fan-in at most $2^{2^{h'}}$, and the *and* gates feeding into g_B have fan-in at most $2^{h'}$, so the weft of the circuit has been preserved by this transformation, which may increase the size of C by the constant factor $2^{2^{h'}}$. The topmost level of large gates (to which the branches B are attached in C) consists of Or gates, in Case 1, so that the gates g_B can be merged into this topmost level. Merging is performed similarly in Case 2, where the replacing circuits are products-of-sums, and the topmost level of large gates consists of And gates. For the next step we consider two cases, depending on whether the topmost level of large gates consists of Or gates or And gates. (Essentially, this corresponds to whether weft t is even or odd.)

Step 6. A monotone change of variables. (Two cases.)

In this step (in both cases) we employ a "change of variables" based on the combinatorial reduction of Theorem 2.1. The goal is to obtain an equivalent circuit that has the property that either all the inputs are MONOTONE (case 1.) (i.e. no inverters in the circuit), or all the inputs are negated with no other inverters in the circuit, which we call ANTIMONOTONE. (Actually in this case we will have *some* of the inputs positive but these will only be $enforcers$ as we will see. So we should call this case NEARLY ANTIMONOTONE.) (case 2.) The point of this step becomes apparent in the next step when we use the special character of the circuit thus constructed to enable us to eliminate the small gates.

Consider the reduction of Theorem 2.1. This reduction consists of two parts. The first is the ring of selection gadgets which allow variable choice, gap choice, and then gap enforcement and the second part is consistency obtained by clause wiring. The idea is to "hard wire" the selection and consistency parts of the construction into the circuit. The point being that we can replace positive instances of variable fanout in the original circuit by outputs corresponding to choice of that variable in the positive selection component. We can replace negative fanout in the original circuit by the appropriate sets of gap variables. Finally we can wire in the fact that we need a dominating set and other enforcements by using the facts that we will only look at a weight $2k$ input, and a And of Or's, which will not add to the $weft$ of the circuit. We argue more precisely below, and also prove in two parts that the whole process cn be accomplished without increasing $weft$, given that the weft is ≥ 2.

Suppose the inputs to the circuit C received at the beginning of this step are $x[1], \ldots, x[n]$, and suppose that the output gate of C is an And gate. Let Y denote the boolean expression having $2n$ clauses, with each clause consisting of a single literal, and with one clause for each

of the $2n$ literals of the n input variables. The reduction of theorem 2.1 allows to translate Y into a monotone formula via dominating set, and thus capturing monotonically all the relevant input settings. Thus, let G_Y be the graph constructed for this expression as in the proof of Theorem 2.1. Note that only part of G_Y will actually be wired into C.

Keeping this in mind, and using the variable (vertex) set obtained from G_Y, the change of variables is implemented for C as follows. (1) Create a new input for each vertex of G_Y that is not a clause vertex. (2) Replace each positive input fanout of $x[i]$ in C with an *Or* gate having k new input variable arguments corresponding to the vertices to which the clause vertex for the clause $(x[i])$ of Y is adjacent in G_Y. (3) Replace each negated fanout line of $x[i]$ with an *Or* gate having $O(n^2)$ new input variable arguments corresponding to the vertices to which the clause vertex for the clause $(\neg x[i])$ of Y is adjacent in G_Y. (4) Merge with the output *And* gate of C a new circuit branch corresponding to the product-of-sums expression, where the product is taken over all non-clause vertices of G_Y, and the sum for a vertex u is the sum of the new inputs corresponding to the non-clause vertices in $N[u]$ (this is the dominating set and other enforcements.).

The modified circuit C' obtained in this way accepts a weight $2k$ input vector if and only if the original circuit C accepts a weight k input vector. The proof of this is essentially the same as for Theorem 2.1. If all of the *not* gates of C are at the top, then the circuit C' will be MONOTONE. However, to see that this change of variables can be employed to obtain a monotone or nearly antimonotone circuit *without* increasing weft, we must consider two cases.

Case 1. The topmost large-gate level consists of *Or* gates.

Let C denote the circuit obtained from step 5 and perform a change of variables as described above. The sequence of transformations of C for this step is shown schematically in Diagram 2.

INSERT DIAGRAM 2 HERE

The result is a circuit C' with no *not* gates. The input weight we are now concerned with is $2k$, and the construction of C' from C may involve quadratic blow-up.

Next, we move the small *and* gates on the second level upward past the *Or* gates introduced by the change of variables, and then merge the *Or* gates down to the topmost 6prs2-gpte layer (of *Or* gates).

Case 2. The topmost large-gate level consists of *And* gates.

Here we use a similar argument, beginning with a trick. Below each gate of the topmost large-gate level (of *And* gates), a double negation is introduced (equivalently). One of the *not* gates is moved to the top of the circuit (by DeMorgan's identities). This is followed by

a change of variables based on Theorem 2.1, as in Case 1. The second level *and* gates are commuted upwards, and the *Or* gates of the second and third levels are merged, as in Case 1. So now the circuit has no negated inputs and no inverters except the residual ones below the top layer of *Or* gates. Finally, these remaining *not* gates are commuted to the top. Note that this means that all fanouts are negated except the ones to the enforcement *Or* gate added during step (4).

We are now in position for the last step.

Step 7. Eliminating the remaining small gates.

If we regard the inputs to C as variables, this step consists of another "change of variables." Let k be the relevant weight parameter value supplied by the last transformation. In this step we will produce a circuit C' corresponding directly to a t-normalized boolean expression (that is, consisting only of t alternating layers of *And* and *Or* gates) such that C accepts a weight k input vector if and only if C' accepts a vector of weight $k' = k \cdot 2^{k+1} + 2^k$.

Suppose that C has m remaining small gates. In Case 1, these are *and* gates, and the inputs are all positive. In Case 2, these are *or* gates, and the inputs to these gates are all negated. For $i = 1, ..., m$ we define the sets A_i of the inputs to C to be the sets of input variables to these small gates. The central idea for this step is to create new inputs representing the sets A_i of inputs to C.

For example, suppose (Case 1) that the output of the small *and* gate g_i in C is the boolean product $(abcd)$ of the inputs a, b, c, d to C. Thus $A_i = \{a, b, c, d\}$. The gate g_i can be eliminated by replacing it with an input line from a new variable $v[i]$ which represents the predicate $a = b = c = d = 1$. (This representation, of course, will need to be enforced by additional circuit structure.) Similarly (Case 2) if g_i computes the value $(\bar{a} + \bar{b} + \bar{c} + \bar{d})$ then g_i can be replaced by a negated input line from $v[i]$.

Let $x[j]$ for $j = 1, ...s$ be the input variables to C. We introduce new input variables of the following kinds:
(1) One new variable $v[i]$ for each set A_i for $i = 1, ..., m$ to be used as indicated above.
(2) For each $x[j]$ we introduce 2^{k+1} copies $x[j, 0], x[j, 1], x[j, 2], ..., x[j, 2^{k+1} - 1]$.
(3) "Padding" consisting of 2^k meaningless variables (inputs not supplying output to any gates) $z[1], ..., z[2^k]$.

We add to the circuit an enforcement mechanism for the change of variables. The necessary requirements can be easily expressed in P-o-S form, and thus can be incorporated into the bottom two levels of the circuit as additional *Or* gates attached to the bottommost (output) *And* gate of the circuit.

We require the following implications concerning the new variables:

(1) The $s \cdot 2^{k+1}$ implications, for $j = 1, ..., s$ and $r = 0, ..., 2^{k+1} - 1$,

$$x[j, r] \Rightarrow x[j, r + 1 \pmod{2^{k+1}}]$$

(2) For each containment $A_i \subseteq A_{i'}$, the implication

$$v[i'] \Rightarrow v[i]$$

(3) For each membership $x[j] \in A_i$, the implication

$$v[i] \Rightarrow x[j, 0]$$

(4) For $i = 1, ..., m$ the implication

$$(\prod_{x[j] \in A_i} x[j, 0]) \Rightarrow v[i]$$

It may be seen that this transformation may increase the size of the circuit by a linear factor exponential in k. We make the following argument for the correctness of the transformation.

If C accepts a weight k input vector, then setting the corresponding copies $x[i, j]$ among the new input variables accordingly, together with appropriate settings for the the new "collective" variables $v[i]$ yields a vector of weight between $k \cdot 2^{k+1}$ and $k \cdot 2^{k+1} + 2^k$ that is accepted by C'. The reason the weight of this corresponding vector may fall short of $k' = k \cdot 2^{k+1} + 2^k$ is that not all of the subsets of the k input variables to C having value 1 may occur among the sets A_i. An accepted vector of weight exactly k' can be obtained by employing some of the "padding" input variables $z[i]$ to C'

Note that the seemingly simpler strategy of creating a new input variable for each set of at most k inputs to C would not serve our purposes, since it would involve increasing the size n of the circuit to possibly n^k. (We are limited in our computational resources for the reduction to $f(k)n^\alpha$. The constant α can be an arbitrary function of the depth and weft bounds h and t, but not k.)

For the other direction, suppose C' accepts a vector of weight k'. Because of the implications in (1) above, exactly k sets of copies of inputs to C must have value 1 in the accepted input vector. Because of the implications (2)–(4), the variables $v[i]$ must have values in the accepted input vector compatible with the values of the sets of copies. By the construction of C', this implies there is a weight k input vector accepted by C. □

Corollary 3.2 (i) For $t > 0$, MONOTONE $W[2t]$= $W[2t]$.
(ii) For $t > 0$ WIEGHTED MONOTONE $2t$-NORMALIZED SATISFIABILITY is $W[2t]$-complete.
(iii) For $t > 0$, MONOTONE $W[2t + 1] = W[2t]$.

Proof (i) The proof come from the analysis of step 6, Case 1.

(ii) We can apply Step 6, Case 1, to a $2t$-normalized formula. The result is a $2t$ normalized monotone formula.

(iii) This result follows by the transformations of Step 7, applied to the formulae supplied by Steps 1-6 applied to formulae in the appropriate classes. □

We remark that in [DF2] we have proven the complementary result for $W[2t+1]$ by showing that $W[2t+1]$ = ANTIMONOTONE $W[2t+2]$. It is an open question whether ANTIMONOTONE $W[2] = W[1]$. The problem is that the relevant gadgets seem to need two levels to enact. Note that the above theorem fails to identify a problem complete for $W[1]$. In the subsequent paper [DF2] we will also show that INDEPENDENT SET, and a number of other natural parameterized problems are complete for $W[1]$. There we show that $W[1] = W[1,2]$ where $W[1,2]$ is equivalent to the problem of given a formula X in conjunctive normal form and of clause size two, does X have a satisfying assignment of weight k? While the unparameterized problem that considers (X,k) is NP- complete (easy reduction from independent set) variations of this are classically P-time or FPT. For instance the problem that asks if there is any satisfyiny assignment is well known to be P-time, and the problem that asks if there is a sayisfying assignment of weight *less than or equal to* k is FPT as follows: Let X be the $2-CNF$ formula and let k be given. If X has no clause without negated literals, done since the assignment with all false works. Otherwise choose some clause C with only positive literals. One of these must be made true so we begin a tree of possibilities. Continue inductively in this way . As the clause size is bounded here by 2 this only gives a factor of 2^k.

The $W[t]$ hierarchy reflects, in a finely resolved way, the difficulty of "solution checking." What happens if, more bluntly, we simply address fixed-parameter complexity for problems for which solutions can be checked in *polynomial* time? To study this question, as we mentioned earlier, it is natural to define the following complexity classes.

Definition. A parameterized problem L belongs to $W[P]$ ($W[SAT]$) (*monotone* $W[P]$) if L uniformly reduces to the parameterized circuit problem L_F for some family of (Boolean) (monotone) circuits F.

Note that $W[t]$ is contained in $W[P]$ for every t, and that $W[P] = FPT$ if $P = NP$. With Karl Abrahamson, [ADF1,2], we have been able to show that all of the problems identified in [AEFM] as complete for PGT are uniformly complete for $W[P]$. (For the reasons mentioned before, we would argue that the present theory offers a better framework for those results, and allows to address much wider parameterized issues.) We have also identified a number of further natural complete problems. These results and some aspects of the structure of $W[P]$ as well as parameterized $PSPACE$ are reported in [ADF1,2]. For $PSPACE$ there are very interesting problems that seem hard and are natural ones that

relate to winning strategies for k-move games.

4. Summary and Open Problems

We have presented in this paper a basic framework and fundamental completeness results for the study of fixed-parameter tractability. We view the exploration of this topic as a large project, of which this constitutes only the initial step.

In some ways, the study of fixed-parameter tractability and completeness addresses the subject of computational infeasibility inside of P. For related work from a different perspective see Buss and Goldsmith [BG] and the references cited there. Many of the approaches and issues concerning the standard complexity classes have natural analogues in this setting that are so far unexplored.

Consider, for example, the issue of parallel complexity. Trivially, there is a parallel algorithm running in time $O(\log n)$ and using n^k processors to determine if a graph G on n vertices has a dominating set of size k, for each fixed k. For a contrasting result, Lagergren [La] has shown that for each fixed k, it can be determined in time $O(\log^3 n)$ with $O(n)$ processors whether a graph has treewidth at most k. This suggests a natural fixed-parameter analogue of NC. Similar remarks apply to randomized complexity.

For another example, consider approximation algorithms. One of the fundamental results of Robertson and Seymour (quite apart from their work on graph minors) is that there is an algorithm that in time $f(k) \cdot n^2$ finds, for a graph G of order n, either: (1) a tree decomposition of width at most $5k$, or (2) evidence that the treeewidth of G is greater than k. An analogous result for DOMINATING SET might be an algorithm running in time $f(k) \cdot n^c$ that finds either: (1) a dominating set of size $O(k)$, or (2) evidence that the minimum size of a dominating set for G is greater than k. Such an algorithm is presently unknown. It may even be that the existence of such an algorithm would imply the collapse of the W hierarchy, much as the existence of a P-time relative approximation algorithm for the Travelling Salesperson problem would imply $P = NP$ [GJ].

Many interesting structural questions concerning the W hierarchy remain to be explored. For instance we have established some connections between classical classes and our parameterized classes. Thus if $W[2t] \subseteq FTP$, then $2t$-NORMALIZED SATISFIABILITY is soluable in time $p(n)2^{o(v)}$, where n is the size of the input, p is a polynomial and v is the number of variables. For this result and other structural results along these lines see [ADF2]. Nevertheless, the precise relationship between classical classes and the parameterized one is still unclear. We also are not aware of an oracle separating the W-hierarchy, nor do we know if collapse propogates upwards. (I.e. if $FPT = W[t]$ implies $FPT = W[t+1]$, for instance.)

From a concrete point of view, we also do not know if a problem such as the following belongs to $W[t]$ for any t.

TWO PLAYER DOMINATING SET
Instance: A graph $G = (V, E)$ and a positive integer k.
Question: Is it true that for every k-element subset $V' \subseteq V$, there is a k-element subset $V'' \subseteq V$ such that $V' \cup V''$ is a $2k$-element dominating set for G?

We have been able to prove the following density theorem.

Theorem 4.1. For the strong uniform reduction hierarchy, if any of the containments

$$FPT \subseteq W[1] \subseteq W[2] \subseteq \cdots$$

is proper, then there are infinitely many intervening equivalence classes of parameterized problems with respect to strong uniform reductions. □

Actually, we can prove a much stronger result along the lines of the full Ladner theorem. Also [DF3] contains quite a number of other structural and relativisation results. For instance we know that there is an oracle with $P \neq NP$ yet the W hierarchy collapses. It is an open question whether an analogue of 4.1 holds in the uniform case. We remark that the setting of parameterized problems introduces some technical challenges for density results. Our proof of Theorem 4.1 (to appear in [DF3]) employs techniques from the infinite-injury priority method.

Lastly, we think the primary value of our theory of fixed-parameter tractability is that there is, for many parameterized problems, a compelling practical interest. There are many natural parameterized problems that may well be complete for various levels of the W hierarchy. Demonstrations of such completeness would provide an explanation of why, though they are solvable in polynomial time for each fixed parameter value, these problems resist attempts to show fixed-parameter tractability.

Acknowledgements Thanks to Karl Abrahamson for useful early discussions about this work, and for suggestions on improving the exposition. Special thanks to Mike Hallett and H. Todd Wareham who prepared the final version of the appendix.

References

[ADF1] K. Abrahamson, R. Downey and M. Fellows, Fixed-Parameter Intractability II, in STACS 93, Springer Verlag, Lecture Notes in Computer Science, (1993) 374-385.

[ADF2] K.Abrahamson, R.Downey, and M.Fellows, Fixed Parameter Tractability and Completeness IV: on Completeness for W[P] and PSPACE Analogues, to appear Annals of Pure and Applied Logic.

[AEFM] K. R. Abrahamson, J. A. Ellis, M. R. Fellows and M. E. Mata, "On the Complexity

of Fixed-Parameter Problems." In *30th Annual Symposium on Foundations of Computer Science*, IEEE Computer Society Press (1989), 210-215.

[AF] K. R. Abrahamson and M. R. Fellows, "Finite Automata, Bounded Treewidth, and Well-quasiordering,"in *Graph Structure Theory* (ed. N. Robertson and P. Seymour) Contemporary Mathematics Vol 147, American Mathematical Society, Providence, RI (1993) 539-564.

[AM] R. Anderson and E. Mayr, "Approximating P–complete problems," Stanford University (1986)

[Ar] S. Arnborg, "Efficient Algorithms for Combinatorial Problems on Graphs with Bounded Decomposability — A Survey," *BIT* 25 (1985), 2-23.

[ACP] S. Arnborg, D. G. Corneil, and A. Proskurowski, "Complexity of finding embeddings in a k-tree", *SIAM Journal on Algebraic and Discrete Methods*, 8:277-284, 1987.

[ACPS] S. Arnborg, B. Courcelle, A. Prokurowski and D. Seese, "An Algebraic Theory of Graph Reduction," Technical Report 90-02, Laboratorie Bordelais de Recherche en Informatique, Universite de Bordeaux I, January 1990.

[ALS] S. Arnborg, J. Lagergren and D. Seese, "Problems Easy for Tree-Decomposable Graphs (extended abstract)." In T. Lepisto and A. Salomaa, eds., *Proc. 15th Int. Coll. Automata, Languages and Programming*, Lecture Notes in Computer Science, Vol. 317 (Springer, Berlin, 1988), 38-51.

[ADP] S. Ausiello, A. D'Atri, and M. Protasi, "Structure Preserving Reductions among Convex Optimization Problems", *Journal of Computer and System Sciences*, 21, 136-153, 1980.

M. W. Bern, E. L. Lawler and A. L. Wong,"Linear Time Computation of Optimal Subgraphs of Decomposable Graphs,"Journal of Algorithms, Vol 8, (1987) 216-135

[BM] D. Bienstock and C. L. Monma, "On the Complexity of Covering Vertices by Faces in a Planar Graph," *SIAM J. Comp.* 17 (1988), 53-76.

[Bod1] H. L. Bodlaender, "On Linear Time Minor Tests and Depth First Search," In F. Dehne et al., eds., *Proc. 1st Workshop on Algorithms and Data Structures*, Lecture Notes in Computer Science, Vol. 382 (Springer, Berlin, 1989), 577-590.

[Bod2] H. L. Bodlaender, "On Disjoint Cycles," Technical Report RUU-CS-90-29, Dept. of Computer Science, Utrecht University, Utrecht, The Netherlands, August 1990.

[Bod3] H. L. Bodlaender, private communication, 1994.

[BDFHW] H. Bodlaender, R. Downey, M. Fellows, M. Hallett, and T. Wareham, "Parameterized Complexity Analysis in Computational Biology", submitted to IEEE Workshop on Shape and Pattern Matching in Computational Biology, held in conjunction with 1994 IEEE Conference on Computer Vision and Pattern Recognition.

[BDFW] H. Bodlaender, R. Downey, M. Fellows, and T. Wareham, "The Parameterized Complexity of Sequence Alignment and Consensus (extended abstract)", to appear, Proceedings of the Fourth Conference on Combinatorial Pattern Matching (CPM '94).

[BDFW2] H. Bodlaender, R. Downey, M. Fellows, and T. Wareham, "The Parameterized Complexity of sequence Alignment and Consensus," submitted Theoretical Computer Science.

[BE] H. Bodlaender and J. Engelfriet, *Domino Treewidth.* Technical Report UU-CS-1994-11, Department of Computer Science, Utrecht University, Utrecht, the Netherlands.

[BFH] H. Bodlaender, M. Fellows, and M. Hallett, "Beyond *NP*-Completeness for Problems of Bounded Width: Hardness for the *W* Hierarchy (Extended Abstract)", to appear *STOC 27.*

[BFW] H. Bodlaender, M. Fellows, and T. Warnow, *Two Strikes Against Perfect Phylogeny.* Technical Report RUU-CS-92-08, Department of Computer Science, Utrecht University, Utrecht, the Netherlands.

[BK] H. Bodlaender and D. Kratsch, private communication, 1994.

[BW1] G. Brightwell and D. Winkler, "Counting linear extensions," Order, 8, 225–242 (1991)

[BW2] G. Brightwell and D. Winkler, " Counting linear extensions is $\#P$–complete," *Proc. 23rd STOC,* (1991) 175–181

[Bu] S. Buss, private communication, 1989.

[BG] S. Buss and J. Goldsmith,"Nondeterminism within P", SIAM Journal of Computing, Vol 22, (1993) 560-572.

[CC1] L. Cai and J. Chen, "On the Amount of Nondeterminism and the Power of Verifying," to appear.

[CC2] L. Cai and J. Chen, "On Input Read Modes of Alternating Turing Machines," submitted.

[CC3] L. Cai and J. Chen, "Fixed Parameter Tractability and Approximability of NP-hard Optimization Problems," to appear

[CCDF1] L. Cai, J. Chen, R. Downey, and M. Fellows, "The Parameterized Complexity of Short Computation and Factorization," to appear *Proceedings of the Sacks Conference, 1993.*

[CCDF2] L. Cai, J. Chen, R. Downey, and M. Fellows,"Advice Classes of Parameterized Tractability,"submitted.

[CCDF3] L. Cai, J. Chen, R. Downey, and M. Fellows,"On the Structure of Parameterized Problems in NP," to appear, STACS 94 (Springer Verlag.)

[Ces] M. Cesati,private communication, 1994.

[Con] A. Condon, "The Complexity of the Max Word Problem and the Power of One Way Interactive Proof Systems," *STACS*, (1991).

[Co] B. Courcelle, "Graph Rewriting: An Algebraic and Logical Approach." In J. van Leeuwen, ed., *Handbook of Theoretical Computer Science, Vol. B* (North-Holland, Amsterdam, 1990), Chapter 5.

[DJPSY] E. Dahlhaus, D. Johnson, C. Papdimitriou, P. Seymour, and M. Yannakakis, "The complexity of multiway cuts," *STOC*, (1992), 241–251.

[Dav] M. Davis, "Unsolvable Problems", in *The Handbook of Mathematical Logic*, J. Barwise (ed.), Elsevier, 1977, p. 580.

[DS] W. Day and D. Sankoff, "Computational Complexity of Inferring Phylogenies by Compatibility", *Systematic Zoology*, 35(2), 224-229, 1986.

[DEF] R. Downey, P. Evans and M. Fellows,"Parameterized Learning Complexity," in *Proceedings of the 6th Annual Conference on Learning Theory, COLT'93* ACM Press, New York, (1993) 51-57.

[DF1] R. Downey and M. Fellows, " Fixed-Parameter Tractability and Completeness," *Congressus Numerantium*, Vol 87 (1992) 161-187.

[DF2] R. Downey and M. Fellows, "Fixed Parameter Tractability and Completeness II: On Completeness for W[1]," to appear Theoretical Computer Science.

[DF3] R. Downey and M. Fellows, "Fixed Parameter Tractability and Completeness III: Density and Other Structural Aspects" in *Complexity Theory* (Ed. K. Ambos-Spies, S. Homer, and U. Schoning) Cambridge University Press (1993) 166-191.

[DF4] R. G. Downey and M. F. Fellows,"Fixed-Parameter Intractability," Proceedings *Structure in Complexity* 7th Annual Conference, IEEE, (1992) 36-49.

[DF5] R. G. Downey and M. F. Fellows,"Parameterized Computational Feasibility,"in *Feasible Mathematics II,* (ed. P. Clote and J. Remmel) Birkhauser, Boston, to appear.

[DF6] R. G. Downey and M. F. Fellows,*Parameterized Complexity* Monograph in preparation.

[DF7] R. G. Downey and M. F. Fellows,"Fixed-Parameter Tractability and Intractability-a Survey," in preparation.

[DFKHW1] R. Downey, M. Fellows, B. Kapron, M. Hallett, and T. Wareham, "Parameterized Complexity of Some Problems in Logic and Linguistics (Extended Abstract)", the 2nd Workshop on Structural Complexity and Recursion-theoretic Methods in Logic Programming, 1993.

[DFKHW2] R. Downey, M. Fellows, B. Kapron, M. Hallett, and T. Wareham,"Parameterized Complexity of Some Problems in Logic and Linguistics," to appear, *Logic at St. Petersberg,* Springer-Verlag, to appear.

[DW] S. Dreyfus and R. Wagner, "The Steiner Problem in Graphs", NETWORKS, 1, 195-207, 1971.

[Dub] E. Dubrova, private communication, 1994.

[ES] M. El–Zahar & J. Schmerl, "On the size of jump critical ordered sets," Order, vol 1, (1984) 3–5.

[EP] P. Erdős and L. Pósa, "On independent circuits contained in a graph," Canad. J. Math., vol 17, (1965) 347-352.

[Fe1] M. Fellows, "The Robertson-Seymour Theorems: A Survey of Applications,"in *Contemporary Mathematics* Vol 89, AMS, (1989) 1-18.

[Fe2] M. Fellows, private communication, 1994.

[FHW] M. Fellows, M. Hallett, and T. Wareham, "DNA Physical Mapping: Three Ways Difficult". In T. Lengauer (ed.) Proceedings: ESA'93 - European Symposium on Algorithms. Lecture Notes in Computer Science no. 726. Springer-Verlag; Berlin. pp. 157-168.

[FK1] M. Fellows and N. Koblitz,"Self-Witnessing Polynomial Time Complexity and Prime Factorization,"in Proceedings *Structure in Complexity* 7th Annual Conference, IEEE, (1992) 107-111.

[FK2] M. Fellows and N. Koblitz, *Fixed-Parameter Complexity and Cryptography*, Technical Report DCS-207-IR, Department of Computer Science, University of Victoria, 1992.

[FL1] M. R. Fellows and M. A. Langston, "On Search, Decision and the Efficiency of

Polynomial-Time Algorithms." In *Proc. Symp. on Theory of Computing (STOC)* (1989), 501-512.

[FL2] M. R. Fellows and M. A. Langston, "On Well-Partial-Order Theory and Its Application to Combinatorial Problems of VLSI Design", *SIAM Journal on Discrete Mathematics*, 5(1), 117–126, 1992.

[FL3] M. R. Fellows and M. A. Langston, "An Analogue of the Myhill-Nerode Theorem and Its Use in Computing Finite Basis Characterizations." In *Proc. Symp. Foundations of Comp. Sci. (FOCS)* (1989), 520-525.

[FL4] M. R. Fellows and M. A. Langston, "Nonconstructive Advances in Polynomial-Time Complexity", *Information Processing Letters*, 28 (1987/88), 157–162.

[FL5] M. R. Fellows and M. A. Langston,"Nonconstructive Tools for Proving Polynomial Time Decidability,"J. Assoc. Comput. Mach., Vol 35,(1988) 727-739.

[GJ] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness* (Freeman, San Francisco, 1979).

[GKR] W. Gasarch, M. Krentel, and K. Rappoport, "OptP as the Normal Behavior of NP-Complete Problems", to appear in *Mathematical Systems Theory.*

[GGKS] P. Goldberg, M. Golumbic, H. Kaplan, and R. Shamir, "Four Three Strikes Against Physical Mapping of DNA (extended abstract)", manuscript.

[GKS] M. Golumbic, H. Kaplan, and R. Shamir, *On the Complexity of DNA Physical Mapping*, Technical Report 271/93, Tel Aviv University.

[Hal] M. Hallett, private communication, 1994.

[HU] J. Hopcroft, and J. Ullman *Introduction to Automata Theory, Languages, and Computation*, Addison-Wesley Publishing Company: Reading, MA, 1979.

[KS] H. Kaplan and R. Shamir, *Pathwidth, Bandwidth and Completion Problems to Proper Interval Graphs with Small Cliques*, Technical Report 285/93, Tel Aviv University.

[La] J. Lagergren, "Algorithms and Minimal Forbidden Minors for Tree-Decomposable Graphs," Dissertation, Department of Numerical Analysis and Computing Science, Royal Institute of Technology, Stockholm, Sweden, March 1991.

[KF] C. Kintala and P. Fischer,"Refining Nondeterminism in Relativised Polynomial Bounded Compuataions," SIAM J. of Comput., Vol 9 (1980) 46-53.

[Lew] H. R. Lewis, "Complexity Results for Classes of Quantificational Formulas ", *Journal*

of Computer and System Sciences, 21 (1980), 317–353.

[Lu] E. Luks, "Isomorphism of graphs of bounded valence can be tested in polynomial time," *JCSS* 25 (1982) 42-65.

[NP] J. Nesetril and S. Poljak, "On the Complexity of the Subgraph Problem," Commen. Math. Univ. Carol. 26 (1985), 415-419.

[PY] C. Papdimitriou and M. Yannakakis, "On Limited Nondeterminism and the Complexity of the V-C Dimension", *Eight Annual Conference on Structure in Complexity Theory*, 12–18, 1993.

[Par] T. D. Parsons, "Pursuit-Evasion in a Graph", in *Theory and Applications of Graphs*, Y. Alavi and D. R. Lick, (eds.), Springer-Verlag, Berlin, 1976, pp. 426–441.

[PM] A. Paz, and S. Moran, "Non Deterministic Polynomial Optimization Problems and their Approximations", *Theoretical Computer Science*, 15, 251-277, 1981.

[Pu] B. Pulleybank, "On minimizing setups in precedence constrained scheduling," *Discrete Appl. Math.,* vol 11 (1985).

[PV] J. Plehn and B. Viogt,"Finding Minimally Weighted Subgraphs," to appear.

[Ris] E. Ristad *Complexity of Simplified Segmental Phonology.* Technical Report CS-TR-388-92 (revised May 1993), Department of Computer Science, Princeton University.

[Re] K. Regan,"Finite Substructure Languages,"in *Proceedings 4th Structure in Complexity Conf.* (1989) 87-96.

[RS1] N. Robertson and P. D. Seymour, "Graph Minors XIII. The Disjoint Paths Problem," to appear.

[RS2] N. Robertson and P. D. Seymour, "Graph Minors XV. Wagner's Conjecture," to appear.

[Sim] H. U. Simon, "Continuous Reductions Among Combinatorial Optimization Problems", *Acta Informatica*, 26, 771–785, 1989.

[VV] L.G.Valiant and V.V.Vazirani, "NP is as easy as detecting unique solutions," *Theoretical Computer Science* 47 (1986), 85-93.

[VL] L. Van Leeuwin, *Handbook of Theoretical Computer Science, Vol. A.* North Holland, 1990.

[VW] D. Vertigan and G. Whittle, "Recognising polymatriods associated with hypergraphs,"

(to appear). Combinatorics, Probability and Computing.

[War] T. Wareham, private communication, 1992–1994.

[WHL] T. V. Wimer, S. T. Hedetniemi and R. Laskar, "A Methodology for Constructing Linear Graph Algorithms," *Congressus Numerantium* 50 (1985), 43-60.

Appendix : A Problem Compendium and Guide to W-Hierarchy Completeness, Hardness and Classification; and Some Open Questions.

This appendix contains problem definitions and summaries of most of the presently known completeness and hardness results, and information concerning fixed-parameter tractability for restrictions of problem instances. References are given, where appropriate. The problems discussed are grouped (more or less) according to level in the W hierarchy. In some cases where membership in a given level is not obvious, the arguments establishing membership are given. Our list for FPT is obviously incomplete, but the given examples should provide the reader with the flavour of such results.

In FPT

Remark. The following is only a small list of the known FPT problems. Many more can be obtained by for instance Courcelle's theorem (see [Co], [AF]) applied to classes of bounded tree width, in conjunction with Bodlaender's theorem on Treewidth k recognition.

Crossing Number for Maximum Degree 3 Graphs
Instance: A graph G all of whose vertices have max degree 3.
Parameter: k
Question: Does G have a an embedding with crossing number $\leq k$?

This is $O(n^3)$ by [Fe1] via [RS1,2].

Alternating Hitting Set
Instance: A collection C of subsets of a set B with $|c| \leq k_1$ for all $c \in C$, an integer k_2.
Parameter: k_1, k_2
Question: Does player one have a forced win in $\leq k_2$ moves in the following game played on C and B? Players alternate choosing a new element of B until, for each $c \in C$, some member of c has been chosen. The player whose choice causes this to happen loses.

The general version of this problem is PSPACE-complete by a reduction from QBF [GJ, GP7]. This problem is in FPT by [ADF2]. Solvable in $O(n)$ time for fixed k_1 and k_2.

Cutwidth
Instance: A graph $G = (V, E)$, an integer k.
Parameter: k
Question: Is the cutwidth of $G \leq k$?

The general version of this problem is NP-complete by a reduction from *Simple Max Cut* [GJ, GT44]. This problem is in FPT by [FL2]. Solvable in $O(n)$ time for fixed k [Bod2].

Diameter Improvement for Planar Graphs
Instance: A planar graph $G = (V, E)$, an integer k.
Parameter: k
Question: Can G be augmented with additional edges in such a way that the resulting graph G' remains planar and the diameter of G' is at most k?

This problem is in FPT by [DF5, RS1]. Solvable in $O(n)$ time for fixed k [Bod2].

Disjoint Paths
Instance: A graph $G = (V, E)$, $s_1, \ldots, s_k \in V$ *start* vertices, $t_1, \ldots, t_k \in V$ *end* vertices.
Parameter: k
Question: Do there exist vertex disjoint paths $P_1, \ldots, P_k$ such that P_i starts at vertex s_i and ends at vertex t_i for $i = 1 \ldots k$?

The general version of this problem is NP-complete by a reduction from *3SAT* [GJ, ND40]. This problem is in FPT by [RS1]. Solvable in $O(n^3)$ time for fixed k.

Feedback Vertex Set
Instance: A graph $G = (V, E)$, an integer k.
Parameter: k
Question: Is there a set U of k vertices of G such that each cycle of G passes through some vertex of U?

The general version of this problem is NP-complete by a reduction from *Vertex Cover* [GJ, GT7]. This problem is in FPT by [DF5, Bod1]. Solvable in $O(n)$ time for fixed k.

Gate Matrix Layout
Instance: A boolean matrix M, an integer k.
Parameter: k
Question: Is there a permutation of the columns of M so that, if in each row we change to $*$ every 0 lying between the row's leftmost and rightmost 1, then no column contains more than k 1's and $*$'s?

This problem is in FPT by [FL4]. Solvable in $O(n)$ time for fixed k [Bod2]. Equivalent to graph pathwidth.

Graph Genus
Instance: A graph $G = (V, E)$, an integer k.
Parameter: k
Question: Does G have genus k?

The general version of this problem is NP-complete [GJ, Update]. This problem is in FPT by [RS1, FL1]. Solvable in time $O(n^3)$ for fixed k by the results of Robertson and Seymour.

Long Cycle
Instance: A graph $G = (V, E)$, an integer k.
Parameter: k
Question: Does G have a cycle of length $\geq k$?

The general version of this problem is NP-complete by a reduction from *Hamiltonian Circuit* [GJ, ND28]. This problem is in FPT by [FL1]. Solvable in $O(n)$ time for fixed k.

Max Leaf Spanning Tree.
Instance: A graph $G = (V, E)$, an integer k.
Parameter: k
Question: Does G have a spanning tree with k or more leaves?

The general version of this problem is NP-complete by a reduction from *Dominating Set* [GJ, ND2]. This problem is in FPT by [DF5, CCDF2, Bod1]. Solvable in $O(n)$ for fixed k.

Minimum Disjunctive Normal Form
Instance: A set $X = \{x_1, x_2, \ldots, x_n\}$ of variables, a set $A \subseteq \{0,1\}^n$ of implicants, a positive interger m.
Parameter: $|A|$
Question: Is there a DNF expression E over X, having no more than m disjuncts, such that E is true for precisely those truth assignments in A and no others?

The general version of this problem is NP-complete by a reduction from *Set Cover* [GJ, LO9]. This problem is in FPT by [Dub]. Solvable in $O(2^{|A|}n)$ time for fixed A.

Minor Order Test
Instance: Graphs $G = (V, E)$ and $H = (V', E')$.
Parameter: H
Question: Is $H \leq_{minor} G$?

The general version of this problem is NP-complete by a reduction from *Hamiltonian Circuit* [GJ, OPEN2]. This problem is in FPT by [RS1]. Solvable in $O(n^3)$ time for fixed k.

Planar Face Cover

Instance: A graph $G = (V, E)$, an integer k.
Parameter: k
Question: Can G be embedded in the plane so that there are k faces which cover all vertices?

The general version of this problem is NP-complete by a reduction from *Vertex Cover* [Fe2]. This problem is in FPT by [BM]. Solvable in $O(n)$ time for fixed k.

Search Number
Instance: A graph $G = (V, E)$, an integer k.
Parameter: k
Question: Are k searchers sufficient to ensure the capture of a fugitive who is free to move with arbitrary speed about the edges of G?

The general version of this problem is NP-complete [Par]. This problem is in FPT by [FL2]. Solvable in $O(n)$ time for fixed k [Bod2].

Steiner Tree
Instance: A graph $G = (V, E)$, a set S of at most k vertices in V, an integer m.
Parameter: k
Question: Is there a set of vertices $T \subseteq V - S$ such that $|T| \leq m$ and $G[S \cup T]$ is connected?

The general version of this problem is NP-complete by a reduction from *Exact Cover* [GJ, ND12; GKR]. This problem is in FPT by [DW]. Soluable in time $O(3^k n + 2^k n^2 + n^3)$ by the Dreyfus-Wagner algorithm [DW] ([War]).

Treewidth
Instance: A graph $G = (V, E)$, an integer k.
Parameter: k
Question: Does G have treewidth k?

The general version of this problem is NP-complete [ACP]. This problem is in FPT by [Bod3]. Solvable in $O(n)$ time for fixed k.

Vertex Cover
Instance: A graph $G = (V, E)$, an integer k.
Parameter: k
Question: Does G have a vertex cover of size $\leq k$? (A *vertex cover* is a set $V' \subseteq V$, $|V'| \leq k$, such that for every edge $(u, v) \in E$, at least u or v is a member of V'.)

The general version of this problem is NP-complete by a reduction from *3SAT* [GJ, GT1]. This problem is in FPT by [DF5, CCDF2, Bu]. Solvable in $O(2^k n)$ or $O(n + k^k)$ time [DF5, Bu].

In FPT(nonuniform)

Graph Linking Number
Instance: A graph $G = (V, E)$, an integer k.
Parameter: k
Question: Can G be embedded into 3-space such that the maximum size of a collection of topologically linked disjoint cycles is bounded by k

This problem is in FPT(nonuniform) by [RS1, FL1]. Solvable in $O(n^3)$ time for fixed k.

In randomized FPT

Bounded Factor Factorization
Instance: An n-bit positive integer N, a positive integer k.
Parameter: k
Question: Is there a prime factor p of N such that $p < n^k$?

This problem is in randomized FPT by [FK1,2].

Linear Extension Count
Instance: A poset $(P, \leq)$, an integer k.
Parameter: k
Question: Does P have at least k linear extensions?

This problem is in randomized FPT by [BW1, BW2]. Not known to be in FPT.

Polynomially Smooth Number
Instance: An n-bit positive integer N, an integer k.
Parameter: k
Question: Is N n^k–smooth, i.e., is every prime divisor of N bounded by n^k?

This problem is in randomized FPT by [FK1,2]. n^k–smoothness of n-digit numbers is a natural number-theoretic property that arises in the study of polynomial-time complexity. For example, the concept plays a central role in the demonstration that primality is in $UP \cap co\text{-}UP$ [FK1,2].

Small Prime Divisor
Instance: An n-bit positive integer N, a positive integer k.
Parameter: k
Question: Does N have a nontrivial divisor less than n^k?

This problem is in randomized FPT by [FK1,2].

t-Threshold Stable Set

Instance: A directed graph $G = (V, A)$, a positive integer k.
Parameter: k
Question: Does G have a stable set of size k? (A *stable set* is a set of vertices $V' \subseteq V$ such that for every vertex v of $V - V'$, there are fewer than t vertices $u \in V'$ with $uv \in A$.)

This problem is W[1]-complete by a reduction from *Independent Set* (k) [DF6]. Complete for $W[1]$ for every $t \geq 2$.

Binary Cladistic Character Compatibility

Instance: A set C of n binary cladistic characters over m objects, an integer k.
Parameter: k
Question: Is there a subset $C' \subseteq C$, $|C'| = k$, such that all pairs of characters in C' are compatible?

The general version of this problem is NP-complete by a reduction from *Clique* [DS]. This problem is W[1]-complete by the same reduction [DS; War]. See [DS] for details about the problem. The unconstrained-character version of this problem is also W[1]-complete [War]. If $k = |C|$, one obtains the problem *Triangulating Coloured Graphs (Perfect Phylogeny).*

Binary Qualitative Character Compatibility

Instance: A set C of n binary qualitative characters over m objects, an integer k.
Parameter: k
Question: Is there a subset $C' \subseteq C$, $|C'| = k$, such that all pairs of characters in C' are compatible?

The general version of this problem is NP-complete by a reduction from *Binary Cladistic Character Compatibility* [DS]. This problem is W[1]-complete by the same reduction [DS; War]. See [DS] for details about the problem. The unconstrained-character version of this problem is W[1]-hard [War].

Clique

Instance: A graph $G = (V, E)$, a positive integer k.
Parameter: k
Question: Is there a set of k vertices $V' \subseteq V$ that forms a complete subgraph of G (that is, a clique of size k)?

The general version of this problem is NP-complete by a reduction from *Vertex Cover* [GJ, GT19]. This problem is W[1]-complete by a reduction from *Independent Set* (k) [DF2]. Fixed-parameter tractable for planar graphs, and for graphs of maximum degree $f(k)$ for

any fixed function f.

Independent Set
Instance: A graph $G = (V, E)$, a positive integer k.
Parameter: k
Question: Is there a set $V' \subseteq V$ of cardinality k, such that $\forall u, v \in V'$, $uv \notin E$?

The general version of this problem is NP-complete by a reduction from *Vertex Cover* [GJ, GT20]. This problem is W[1]-complete by a reduction from *Antimonotone W[1,2]* [DF2]. Fixed-parameter tractable for planar graphs.

Longest Common Subsequence
Instance: A set of k strings $X_1, \ldots, X_k$ over an alphabet Σ, a positive integer m.
Parameter: k, m
Question: Is there a string $X \in \Sigma^*$ of length at least m that is a subsequence of X_i for $i = 1, \ldots, k$?

The general version of this problem is NP-complete by a reduction from *Vertex Cover* [GJ, SR10]. This problem is W[1]-complete by a reduction from *Clique* (k) [BDFW].

Max Degree d Red/Blue Nonblocker
Instance: A graph $G = (V, E)$ of maximum degree d where V is partitioned into two color classes $V = V_{\text{red}} \cup V_{\text{blue}}$, a positive integer k.
Parameter: k
Question: Is there is a set of red vertices $V' \subseteq V_{\text{red}}$ of cardinality k such that every blue vertex has at least one neighbor that does not belong to V'?

This problem is W[1]-complete by a reduction from *W[1, s] Circuit Satisfiability* [DF2].

Semigroup Embedding
Instance: A semigroup $(S, \cdot)$ and a semigroup $(H, \times)$.
Parameter: H
Question: Can H be embedded into S?

This problem is W[1]-complete by a reduction from *Clique* (k) [DF6,7].

Semilattice Embedding
Instance: A semilattice L and a semilattice H.
Parameter: H
Question: Is H embeddable into L?

This problem is W[1]-complete by a reduction from *Clique* (k) [DF6,7].

Set Packing

Instance: A finite family of sets $S = S_1, ..., S_n$, an integer k.
Parameter: k
Question: Does S contain a subset R of k mutually disjoint sets?

The general version of this problem is NP-complete by a reduction from *X3C* [GJ, SP3]. This problem is W[1]-complete by a reduction from *Independent Set* (k) [ADP; War].

Short Context-Sensitive Derivation
Instance: A context-sensitive grammar $G = (N, \Sigma, \Pi, S)$, a word $x \in \Sigma^*$, an integer k.
Parameter: k
Question: Is there a G-derivation of x of length at most k?

The general version of this problem is PSPACE-complete by a reduction from *Linear Bounded Automaton Acceptance* [GJ, AL20]. This problem is W[1]-complete by a reduction from *Clique* (k) [DFKHW].

Short NDTM Computation
Instance: A nondeterministic Turing machine M operating on alphabet Σ, a word $x \in \Sigma^*$, a positive integer k.
Parameter: k
Question: Is there a computation of M on input x that reaches an accept state in at most k steps?

The general version of this problem is undecidable [HU]. This problem is W[1]-complete by a reduction from *Clique* (k) [DFKHW]. In FPT if either the size of the alphabet or the number of nondeterministic transition possibilities out of a given state is bounded.

Short NDTM Computation
Instance: A nondeterministic Turing machine M operating on alphabet Σ, a word $x \in \Sigma^*$, a positive integer k.
Parameter: k, $|x|$
Question: Is there a computation of M on input x that reaches an accept state in at most k steps?

The general version of this problem is undecidable [HU]. This problem is W[1]-complete by a reduction from *Clique* (k) [DFKHW; Ces].

Short Post Correspondence
Instance: A Post system Π, a positive integer k.
Parameter: k
Question: Is there a length k solution for Π?

This problem is W[1]-complete by a reduction from *Short Unrestricted Grammar Derivation* (k) [CCDF1].

Short Unrestricted Grammar Derivation
Instance: An unrestricted phrase-structure grammar G, a word x, a positive integer k.
Parameter: k
Question: Is there a G-derivation of x of length at most k?

The general version of this problem is undecidable [HU]. This problem is W[1]-complete by a reduction from *Clique* (k) [CCDF1].

Square Tiling
Instance: A set C of "colours", a collection $T \subseteq C^4$ of "tiles" (where $\langle a, b, c, d\rangle$ denotes a tile whose top, right, bottom, and left sides are colored a, b, c, and d, respectively), a positive integer $k \leq C$.
Parameter: k
Question: Is there a tiling of an $k \times k$ square using the tiles in T, i.e., an assignment of a tile $A(i,j) \in T$ to each ordered pair i, j, $1 \leq i \leq k$, $1 \leq j \leq k$, such that (1) if $f(i,j) = \langle a, b, c, d\rangle$ and $f(i+1, j) = \langle a', b', c', d'\rangle$, then $a = c'$, and (2) if $f(i,j) = \langle a, b, c, d\rangle$ and $f(i, j+1) = \langle a', b', c', d'\rangle$, then $b = d'$.

The general version of this problem is NP-complete by a reduction from *Directed Hamiltonian Path* [GJ, GP13]. This problem is W[1]-complete by [CCDF1, Lew].

Vapnik–Chervonenkis (VC) Dimension
Instance: A family of subsets F of a base set X, an integer k.
Parameter: k
Question: Is the VC dimension of F at least k? (The VC dimension of a family of subsets F of a base set X is the maximum cardinality of a set $S \subseteq X$ such that for each subset $S' \subseteq S$, $\exists Y \in F$ such that $S \cap Y = S'$.)

The general version of this problem is LOGSNP-complete [PY]. This problem is W[1]-complete by a reduction from *Clique* (k) [DEF]. Membership of W[1] is proven by a generic reduction in [DF5].

Weighted q-CNF Satisfiability
Instance: A q-CNF formula X, i.e., a CNF formula such that each clause has no more than q literals, an integer k.
Parameter: k
Question: Does X have a satisfying assignment of weight k?

This problem is W[1]-complete by a reduction from *Independent Set* (k) [DF2].

W[1]-Hard, In W[2]

Perfect Code
Instance: A graph $G = (V, E)$, a positive integer k.
Parameter: k
Question: Does G have a k-element perfect code? (A *perfect code* is a set of vertices $V' \subseteq V$ with the property that for each vertex $v \in V$ there is precisely one vertex in $N[v] \cap V'$.)

This problem is W[1]-hard by a reduction from *Independent Set* (k) and in W[2] by [DF2]. We believe that it may be of difficulty intermediate between $W[1]$ and $W[2]$.

Weighted Exact CNF Satisfiability
Instance: A boolean expression E in conjunctive normal form, a positive integer k.
Parameter: k
Question: Is there a truth assignment of weight k to the variables of E that makes exactly one literal in each clause of E true?

This problem is W[1]-hard by a reduction from *Perfect Code* (k) and in W[2] by [DF2]. Equivalent to Perfect Code [DF2]. A related problem is **Unique Weighted Satisfiability** below.

Unique Weighted Satisfiability
Instance: A boolean CNF formula X and an integer k.
Parameter: k.
Question: Is there a unique weight k satisfying assignment for X?

Clearly this is in $W[2]$. There are obvious versions of the above for normalized satisfiability at any level of the W hierarchy. For the CNF situation above, the problem is clearly in the natural analogue of D_p. The reader should recall that D_p consists of the class of languages L that can be expressed as the intersection of a language in NP and one in $co-NP$. Clearly we can similarly define $D_p[2]$ (or, more generally $D_p[t]$) as for D_p but with $W[2]$ in place of NP. Then this problem is in $D_p[2]$, as we see below. Before we prove this we remark that using the Valiant-Vasirani technique([VV]) we can show that UNIQUE WEIGHTED CNF SATISFIABILITY for arbitary boolean formulae is the same as WEIGHTED CNF SATISFIABILITY for boolean formulae under randomized reductions. However, on the face of it, it is not clear if this is true for any finite level of the W hierarchy since for instance weight is lost in the [VV] proof. Nevertheless using a new argument the authors [unpubl.] have shown that a weighted version of [VV] holds for all $t \geq 2$. This seems a very fruitful area to analyse.

UNIQUE weighted CNF satisfiability is in $D_p[2]$

It suffices to describe how to say that a CNF expression has at least two satisfying expressions in $W[2]$. Let C be the circuit corresponding to X. Take two copies of C. Add

$O|X|$ many gates to express the fact that the first copy of C has a satisfying assignment different from the second. Now for k choose q and r appropriately and take q copies of the the left circuit and r copies of the right. Add new gates to express the fact that the inputs of the left q must all be equal and the fact that the inputs of the right r must all be equal. Now accept C if the new circuit has a weight $(q+r)k$ accepting input. Then for the correct choice of (q,r), depending only on k, C has two or more accepting inputs iff the new circuit has one of weight $(q+r)k$. □.

We remark that this seems where UNIQUE DOMINATING SET would lie. We do not at present know if there are $D_p[t]$ complete problems. The analogue for D_p would be a parameterized unique travelling salesperson type problem.

W[1]-Hard, In W[P]

Permutation Group Factorization
Instance: A set A of permutations $A \subseteq S_n$, $x \in S_n$, an integer k.
Parameter: k
Question: Does x have a factorization of length k over A?

This problem is W[1]-hard by a reduction from *Perfect Code* (k) and in W[P] by [CCDF1].

Subset Sum
Instance: A set of integers $X = \{x_1, \ldots, x_n\}$, integers s, k,
Parameter: k
Question: Is there a subset $X' \subseteq X$ of cardinality k such that the sum of the integers in X' equals s?

The general version of this problem is NP-complete by a reduction from *Partition* [GJ, SP13]. This problem is W[1]-hard by a reduction from *Perfect Code* (k) [DF2, FK1,2] and in W[P] by [FK1,2]. Not known to belong to $W[t]$ for any t, but easily placed in $W[P]$.

W[1]-Hard

α-Balanced Separator
Instance: A graph $G = (V, E)$, an integer k.
Parameter: k
Question: Does there exist a set of vertices S, $|S| \leq k$, such that every component of $G[V - S]$ has at most $\alpha|V|$ vertices?

This problem is W[1]-hard by a reduction from *Clique* (k) [BK]. This problem is W[1]-

hard for every fixed α.

Colored Proper Interval Graph Completion
Instance: A graph $G = (V, E)$, a vertex coloring $c : V \to \{1, \ldots, k\}$
Parameter: k
Question: Does there exist a proper interval supergraph of G which respects c?

This problem is W[1]-hard by a reduction from *Independent Set* (k) [KS].

Colored Unit Interval Graph Completion
Instance: A graph $G = (V, E)$, a vertex coloring $c : V \to \{1, \ldots, k\}$
Parameter: k
Question: Does there exist a graph $G' = (V, E')$ such that $E' \supseteq E$, G' is a unit interval graph and G' is properly colored by c?

The general version of this problem is NP-complete [KS]. This problem is W[1]-hard by [KS].

Exact Cheap Tour
Instance: A weighted graph $G = (V, E)$, a weight function $w : E \to \mathbf{Z}$, an integer k.
Parameter: k
Question: Is there a tour through at least k nodes of G of cost exactly S?

The general version of this problem is NP-complete by a reduction from *Hamiltonian Circuit* [GJ, ND22]. This problem is W[1]-hard by [DF6]. See the related *Short Cheap Tour* problem.

INVMAX
Instance: A circuit C, an initial configuration $conf_0$ describing the placing of inverters on connections between gates in G, an integer k.
Parameter: k
Question: Is there a subset A of the gates in G to which DeMorgan's rules can be applied such that the resulting circuit will have at least k gates without any inverters attached to their output lines?

The general version of this problem is NP-complete by a reduction from *Independent Set* [Sim]. This problem is W[1]-hard by the same reduction [Sim].

Proper Interval Sandwich With Bounded Clique Size
Instance: A sandwich instance $S = (V, E^1, E^3)$, an integer k.
Parameter: k
Question: Does there exist a sandwich G for S which is a proper interval graph such that the size of largest clique is at most k?

The general version of this problem is NP-complete [KS]. This problem is W[1]-hard by a reduction from *Colored Proper Interval Graph Completion* (k) [KS].

Reachability Distance for Vector Addition Systems (Petri Nets)
Instance: A set T of m length n integer-valued vectors $T = \{x^i = (x_1^i, \ldots, x_n^i) : 1, \leq i \leq m\}$, a non-negative starting vector $s = (s_1, \ldots, s_n)$, a non-negative target vector $t = (t_1, \ldots, t_n)$, a positive integer k.
Parameter: k
Question: Is there a choice of k indices $i_1, \ldots, i_k$, $1 \leq i_j \leq m$ for $j = 1, \ldots, k$ such that $t = s + \sum_{j=1}^{k} x^{i_j}$ and such that every intermediate sum is non-negative in each component, that is, $s_r + \sum_{j=1}^{q} x_r^{i_j} \geq 0$ for $q = 1, \ldots, k$ and $r = 1, \ldots, n$?

This problem is W[1]-hard by a reduction from *Clique* (k) [DFKHW].

Short l-Tape NDTM Computation
Instance: An l-tape nondeterministic Turing machine M operating on alphabet Σ, a word $x \in \Sigma^*$, a positive integer k.
Parameter: k
Question: Is there a computation of M on input x that reaches an accept state in at most k steps?

The general version of this problem is undecidable [HU]. This problem is W[1]-hard by a reduction from *Short NDTM Computation* (k) [Ces].

Short l-Tape NDTM Computation
Instance: An l-tape nondeterministic Turing machine M operating on alphabet Σ, a word $x \in \Sigma^*$, a positive integer k.
Parameter: k, l
Question: Is there a computation of M on input x that reaches an accept state in at most k steps?

The general version of this problem is undecidable [HU]. This problem is W[1]-hard by a reduction from *Short NDTM Computation* (k) [Ces].

Subset Product
Instance: A set of integers $X = \{x_1, \ldots, x_n\}$, integers a, m, k,
Parameter: k
Question: Is there a subset $X' \subseteq X$ of cardinality k such that the product of the integers in X' is congruent to a mod m?

The general version of this problem is NP-complete by a reduction from *X3C* [GJ, SP14]. This problem is W[1]-hard by a reduction from *Perfect Code* (k) [FK1,2].

Coloured Graph Automorphism

Instance: A 2-coloured (bipartite) graph G
Parameter: k
Question: Is there an automorphism preserving colours moving exactly k blue vertices?

W[2]-Complete

Dominating Set

Instance: A graph $G = (V, E)$, a positive integer k.
Parameter: k
Question: Is there a set of k vertices $V' \subseteq V$ with the property that every vertex of G either belongs to V' or has a neighbor in V'?

The general version of this problem is NP-complete by a reduction from *Vertex Cover* [GJ, GT2]. This problem is W[2]-complete by a reduction from *Weighted Satisfiability* (k) [DF0, DF1]. Fixed-parameter tractable for planar graphs [DF5]. Problem is W[2]-hard if the dominating set V' is required to be either connected or *total*, i.e., for each vertex in V there is an edge to some vertex in V' [BK]. Problem is harder than *Perfect Code*, i.e. W[1]-hard, if a general, connected, or total dominating set is also required to be *efficient*, i.e. each vertex not in V' is dominated by exactly one vertex in V' [BK].

Hitting Set

Instance: A finite family of sets $S = S_1, \ldots, S_n$ comprised of elements from $U = \{u_1, \ldots, u_m\}$, an integer k.
Parameter: k
Question: Is there a subset $T \subseteq U$ of size k such that for all $S_i \in S$, $S_i \cap T \neq \emptyset$?

The general version of this problem is NP-complete by a reduction from *Vertex Cover* [GJ, SP8]. This problem is W[2]-complete by a reduction from *Set Cover* (k) [ADP; War].

Independent Dominating Set

Instance: A graph $G = (V, E)$, a positive integer k.
Parameter: k
Question: Is there a set of k vertices $V' \subseteq V$ that is both an independent set and a dominating set in G?

The general version of this problem is NP-complete by a reduction from *Vertex Cover* [GJ, GT2]. This problem is W[2]-complete by [DF6]. Fixed-parameter tractable for planar graphs.

Set Cover

Instance: A finite family of sets $S = S_1, ..., S_n$, an integer k.
Parameter: k

Question: Is there a subset $R \subseteq S$ whose union is all elements in the union of S?

The general version of this problem is NP-complete by a reduction from *X3C* [GJ, SP5]. This problem is W[2]-complete by a reduction from *Dominating Set* (k) [PM; War].

Tournament Dominating Set
Instance: A tournament T, a positive integer k.
Parameter: k
Question: Does T have a dominating set of cardinality at most k?

The general version of this problem is LOGSNP-complete [PY]. This problem is W[2]-complete by a reduction from *Dominating Set* (k) [DF5].

Weighted $\{0,1\}$ Integer Programming
Instance: A binary matrix A, a binary vector $\mathbf{b}$, an integer k.
Parameter: k
Question: Does $A \cdot \mathbf{x} \geq \mathbf{b}$ have a binary solution of weight k?

The general version of this problem is NP-complete by a reduction from *3SAT* [GJ, MP1]. This problem is W[2]-complete by a reduction from *Monotone Weighted Satisfiability* (k) [DF1]. The problem *Weighted Exact* $\{0,1\}$ *Integer Programming* which asks that equality hold is hard for $W[1]$ [DF6].

W[2]-Hard, In W[P]

Monochrome Cycle Cover
Instance: An edge-colored graph $G = (V, E)$, a positive integer k.
Parameter: k
Question: Is there a set of k vertices $V' \subseteq V$ with the property that every monochrome cycle in G contains a vertex in V'?

This problem is W[2]-hard and in W[P] by [DF6]. Not known to belong to W[t] for any t, but easily shown to be in W[P].

Monoid Factorization
Instance: A set A of self-maps on $[n]$, a self-map h, an integer k.
Parameter: k
Question: Is there a factorization of h of length k over A?

This problem is W[2]-hard by a reduction from *Dominating Set* (k) and in W[P] by [CCDF].

Dominating Clique
Instance: A graph $G = (V, E)$, a positive integer k.
Parameter: k
Question: Is there a set of k vertices $V' \subseteq V$ that forms a complete subgraph of G and is also a dominating set for G?

This problem is W[2]-hard by a reduction from *Dominating Set* (k) [BK]. Problem is in FPT if V' is also required to be *efficient*, i.e. each vertex not in V' is dominated by exactly one vertex in V' [BK].

Longest Common Subsequence
Instance: A set of k strings $X_1, \ldots, X_k$ over an alphabet Σ, a positive integer m.
Parameter: m
Question: Is there a string $X \in \Sigma^*$ of length at least m that is a subsequence of X_i for $i = 1, \ldots, k$?

The general version of this problem is NP-complete by a reduction from *Vertex Cover* [GJ, SR10]. This problem is W[2]-hard by a reduction from *Dominating Set* (k) [BDFW].

Maximal Irredundant Set
Instance: A graph $G = (V, E)$, a positive integer k.
Parameter: k
Question: Is there a set $V' \subseteq V$ of cardinality k such that (1) each vertex $u \in V'$ has a private neighbor and (2) V' is not a proper subset of any $V'' \subseteq V$ which also has this property? (A *private neighbor* of a vertex $u \in V'$ is a vertex u' (possibly $u' = u$) with the property that for every vertex $v \in V'$, $u \neq v$, $u' \notin N[v]$.)

This problem is W[2]-hard by a reduction from *Dominating Set* (k) [BK].

Precedence Constrained k-Processor Scheduling
Instance: A set T of unit-length tasks, a partial order $\prec$ on T, a positive integer deadline D, a number of processors k.
Parameter: k
Question: Is there a map $f : T \to \{1, \ldots, D\}$, such that for all $t, t' \in T$, $t \prec t'$ implies $f(t) < f(t')$, and for all i, $1 \leq i \leq D$, $|f^{-1}(i)| \leq k$?

The general version of this problem is Open [GJ, OPEN8]. This problem is W[2]-hard by a reduction from *Dominating Set* (k) [BFH].

Steiner Tree
Instance: A graph $G = (V, E)$, a set S of at most k vertices in V, an integer m.

Parameter: m
Question: Is there a set of vertices $T \subseteq V - S$ such that $|T| \leq m$ and $G[S \cup T]$ is connected?

The general version of this problem is NP-complete by a reduction from *Exact Cover* [GJ, ND12; GKR]. This problem is W[2]-hard by a reduction from *Dominating Set* (k) [BK].

W[3]-Hard, In W[4]

Dominating Threshold Set
Instance: A graph $G = (V, E)$, integers k, r.
Parameter: k, r
Question: Is there a set $V' \subseteq V$ of at most k vertices such that for every vertex u, $N[u]$ contains at least r elements of V'?

This problem is W[3]-hard and in W[4] by [Fel].

W[t]-Complete

Weighted t-Normalized Satisfiability
Instance: A t-normalized boolean expression X, a positive integer k.
Parameter: k
Question: Does X have a satisfying truth assignment of weight k?

This problem is W[t]-complete by [DF1].

$\leq k$ t-Normalized Satisfiability
Instance: A boolean formula X, an integer k.
Parameter: k
Question: Does X have a satisfying assignment of weight $\leq k$?

For $t \geq 2$ this problem is W[t]-complete by [CC1-3]. For $t = 1$ the problem is in FPT.

W[t]-Hard, For All t, In W[P]

Short Phonological Segmental Decoding
Instance: An integer k, a simplified segmental grammar $s = (F, S, D, R, c_p, C)$ such that the number of mutually exclusive rule sets in R, $|R_{m.e.}|$, is at most k, string $s \in S^+$.
Parameter: k
Question: Is there a string $u \in D$ such that $g(u) = s$?

The general version of this problem is NP-complete by a reduction from *Clique* [Ris]. This problem is W[t]-hard by a reduction from *Weighted t-Normalized Satisfiability* (k) and in W[P] by [DFKHW]. See [DFKHW] and [Ris] for definitions of simplified segmental grammars. The same proof also implies the W[t]-hardness and membership in W[P] of *Short Phonological Segmental Encoding* .

W[t]-Hard, For All t.

Bandwidth

Instance: A graph $G = (V, E)$, a positive integer k.
Parameter: k
Question: Is there a 1:1 linear layout $f : V \to \{1, \ldots, |V|\}$ such that $uv \in E$ implies $|f(u) - f(v)| \leq k$?

The general version of this problem is NP-complete by a reduction from *3-Partition* [GJ, ND40]. This problem is W[t]-hard by a reduction from *Uniform Emulation on a Path* (k) [BFH]. Remains W[t]-hard for all t when given graph is directed and layout must respect arc direction, or when given graph is a tree [BFH]. The related problem *cutwidth* is FPT [FL1].

Colored Cutwidth

Instance: A graph $G = (V, E)$, an edge coloring $c : E \to \{1, \ldots, r\}$, a positive integer k.
Parameter: k
Question: Is there a 1:1 linear layout $f : V \to \{1, \ldots, |V|\}$ such that for each color $j \in \{1, \ldots, k\}$ and for each i, $1 \leq i \leq |V| - 1$, we have $|\{uv : c(uv) = j \text{ and } f(u) \leq i \text{ and } f(v) \geq i + 1\}| \leq r$?

This problem is W[t]-hard by a reduction from *Longest Common Subsequence* (k) [BFH].

Domino Treewidth

Instance: A graph $G = (V, E)$, an integer k.
Parameter: k
Question: Is the domino treewidth of G at most k?

The general version of this problem is NP-complete by a reduction from *Longest Common Subsequence* [BE]. This problem is W[t]-hard by the same reduction [BE].

Feasible Register Assignment

Instance: A directed acyclic graph $G = (V, E)$, a positive integer k, a register assignment $r : V \to \{R_1, \ldots, R_k\}$.
Parameter: k
Question: Is there a linear ordering f of G, and a sequence $S_0, S_1, \ldots, S_{|V|}$ of subsets of V, such that $S_0 = \emptyset$, $S_{|V|}$ contains all vertices of in-degree 0 in G, and for all i, $1 \leq i \leq |V|$,

$f^{-1}(i) \in S_i$, $S_i - \{f^{-1}(i)\} \subseteq S_{i-1}$ and S_{i-1} contains all vertices u for which $(f^{-1}(i), u) \in E$, and for all j, $1 \leq j \leq k$, there is at most one vertex $u \in S_i$ with $r(u) = R_j$?

The general version of this problem is NP-complete by a reduction from *3SAT* [GJ, PO2]. This problem is W[t]-hard by a reduction from *Longest Common Subsequence* (k) [BFH].

Intervalizing Colored Graphs (DNA Physical Mapping)
Instance: A graph $G = (V, E)$, vertex coloring $c : V \rightarrow \{1, \ldots, k\}$.
Parameter: k
Question: Does there exist a supergraph $G' = (V, E')$ where $E \subseteq E'$ and G' is properly colored by c and is an interval graph?

The general version of this problem is NP-complete by a reduction from *Betweenness; Independent Set* [GKS, FHW]. This problem is W[t]-hard by a reduction from *Colored Cutwidth* (k) [BFH]. No polynomial time algorithm is known for fixed k.

Longest Common Subsequence 2
Instance: A set of k strings $X_1, \ldots, X_k$ over an alphabet Σ, a positive integer m.
Parameter: k
Question: Is there a string $X \in \Sigma^*$ of length at least m that is a subsequence of X_i for $i = 1, \ldots, k$?

The general version of this problem is NP-complete by a reduction from *Vertex Cover* [GJ, SR10]. This problem is W[t]-hard by a reduction from *Monotone t-Normalized Weighted Satisfiability* (k) [BFH].

Longest Common Subsequence 3
Instance: A set of k strings $X_1, \ldots, X_k$ over an alphabet Σ, a positive integer m.
Parameter: k, $|\Sigma|$
Question: Is there a string $X \in \Sigma^*$ of length at least m that is a subsequence of X_i for $i = 1, \ldots, k$?

The general version of this problem is NP-complete by a reduction from *Vertex Cover* [GJ, SR10]. This problem is W[t]-hard by a reduction from *Longest Common Subsequence* (k) [BDFHW].

Proper Interval Graph Completion Problem with Minimum Clique Size
Instance: A graph $G = (V, E)$, an integer k.
Parameter: k
Question: Does there exist a graph $G' = (V, E')$, $E' \supseteq E$, such that G' is an interval graph and has minimum clique size k?

The general version of this problem is NP-complete [KS]. This problem is W[t]-hard by

[KS]. Equivalent to the *Bandwidth* problem [KS] ([Hal]).

Triangulating Colored Graphs (Perfect Phylogeny)
Instance: A graph $G = (V, E)$, vertex coloring $c : V \to \{1, \ldots, k\}$.
Parameter: k
Question: Does there exist a supergraph $G' = (V, E')$ where $E \subseteq E'$ and G' is properly colored by c and G' is triangulated?

The general version of this problem is NP-complete by a reduction from *Independent Set* [BFW]. This problem is W[t]-hard by a reduction from *Longest Common Subsequence* (k) [BFH].

Uniform Emulation on a Path
Instance: A graph $G = (V, E)$, an integer k.
Parameter: k
Question: Does there exist a function $f : V \to \{1, \ldots, |V|/k\}$ such that for all $uv \in E$ implies $|f(u) - f(v)| \leq 1$ and for all i, $|f^{-1}(i)| \leq k$?

This problem is W[t]-hard by a reduction from *Monotone t-Normalized Weighted Satisfiability* (k) [BFH]. Remains Wt]-hard for all t when given graph is a tree [BFH].

W[P]-Complete

k-Based Tiling
Instance: A tiling system with distinguished tiles, an integer k.
Parameter: k
Question: Is there a tiling of the $n \times n$ plane using the tiling system and starting with exactly k distinguished tiles in a line ?

This problem is W[P]-complete by [DF4]. Reduction consists of a generic simulation of a Turing machine.

k-Induced 3CNF Satisfiability
Instance: A 3CNF formula φ, an integer k.
Parameter: k
Question: Is there a set of k variables and a truth table assignment to those variables that causes φ to unravel?

This problem is W[P]-complete by a reduction from *Chain Reaction Closure* (k) [ADF2].

k-Induced Satisfiability
Instance: A boolean formula φ, an integer k.
Parameter: k

Question: Is there a set of k variables and a truth table assignment to those variables that causes φ to unravel?

This problem is W[P]-complete by a reduction from *k-Induced 3CNF Satisfiability* (k) [ADF2].

Chain Reaction Closure
Instance: A directed graph $D = (V, A)$, an integer k.
Parameter: k
Question: Does there exist a set V' of k vertices of D whose chain reaction closure is D? (A *chain reaction closure of* V' is the smallest superset S of V' such that if $u, u' \in S$ and arcs ux, $u'x$ are in D then $x \in S$.)

This problem is W[P]-complete by a reduction from *Weighted Monotone Circuit Satisfiability* (k) [ADF2].

Degree 3 Subgraph Annihilator
Instance: A graph $G = (V, E)$, a positive integer k.
Parameter: k
Question: Is there a set of k vertices $V' \subseteq V$ such that $G - V'$ has no subgraph of minimum degree 3?

This problem is W[P]-complete by a reduction from *Weighted Monotone Circuit Satisfiability* (k) [ADF2].

Linear Inequalities
Instance: A system of linear inequalities, an integer k.
Parameter: k
Question: Can we delete k of the equalities and get a system that is consistent over the rationals?

This problem is W[P]-complete by a reduction from *Weighted Circuit Satisfiability* (k) [AEFM, ADF2].

Minimum Axiom Set
Instance: A finite set S of sentences, an implication relation R consisiting of pairs (A, t) where $A \subseteq S$ and $t \in S$, an integer k.
Parameter: k
Question: Is there a set $S_0 \subseteq S$ with $|S_0| \leq k$ and a positive integer n such that if we define S_i, $1 \leq i \leq n$, to consist of exactly those $t \in S$ for which either $t \in S_{i-1}$ or there exists a set $U \subseteq S_{i-1}$ such that if $(U, t) \in R$ then $S_n = S$?

The general version of this problem is NP-complete by a reduction from *X3C* [GJ, L017]. This problem is W[P]-complete by a reduction from *Weighted Circuit Satisfiability*

(k) [DFKHW, ADF2].

Short Circuit Satisfiability
Instance: A boolean circuit C with n gates and at most $k \log n$ inputs and one output.
Parameter: k
Question: Is there a setting of the inputs that cause C to output 1?

This problem is W[P]-complete by a reduction from *Weighted Circuit Satisfiability* (k) [AEFM, ADF2].

Short Satisfiability
Instance: A formula φ on n variables, a list of at most $k \log n$ variables of φ.
Parameter: k
Question: Is there any setting of the distinguished varaibles that causes φ to unravel?

This problem is W[P]-complete by a reduction from *Short Circuit Satisfiability* (k) [ADF2].

Threshold Starting Set
Instance: A directed graph $D = (V, A)$, a positive integer k.
Parameter: k
Question: Does G have a *starting set* of size k? (A *starting set* is a set of vertices $V' \subseteq V$ with the property that if we begin with a pebble on each of the vertices in V' and subsequently place pebbles on any vertex having at least t incoming arcs from pebbled vertices then eventually every vertex of the graph is pebbled.)

This problem is W[P]-complete by a reduction from *Weighted Monotone Circuit Satisfiability* (k) [ADF2].

Weighted Circuit Satisfiability
Instance: A boolean circuit C, a positive integer k.
Parameter: k
Question: Is there a weight k input vector accepted by C?

Weighted Monotone Circuit Satisfiability
Instance: A boolean monotone circuit C, a positive integer k.
Parameter: k
Question: Is there a weight k input vector accepted by C?

This problem is W[P]-complete by a reduction from *Minimum Axiom Set* (k) [DFKHW, ADF2].

Weighted Planar Circuit Satisfiability
Instance: A planar decision circuit C, an integer k.

Parameter: k
Question: Does C have a satisfying assignment of weight k?

This problem is W[P]-complete by a reduction from *Weighted Circuit Satisfiability* (k) [ADF2].

Comment The following classes are not discussed in this paper but are listed for completeness. They are analogues of the QBFSAT and PSPACE in some sense and correspond as we see to the complexity of k move games. They are discussed at length in [ADF1,2]. In each case the first problem defines the class.

AW[t]-Complete. Note that for $t \geq 2$, AW[2]=AW[t] as is proven in [ADF2]

Parameterized QBFSAT$_t$

Instance: An integer r, a sequence $s_1, \ldots, s_r$ of pairwise disjoint sets of boolean variables, a boolean formula X involving the variables $s_1 \cup \ldots \cup s_r$ which consists of t alternating layers of conjunctions and disjunctions with negations applied only to variables, integers $k_1, \ldots, k_r$.
Parameter: r, $k_1, \ldots, k_r$
Question: Is it the case that there exists a size k_1 subset t_1 of s_1 such that for every size k_2 subset t_2 of s_2 there exists a size k_3 subset t_3 of s_3 such that ... (alternating quantifiers) such that, when the variables in $t_1 \cup \ldots \cup t_r$ are made true and all other variables are made false, formula X is true?

Unitary Parameterized QBFSAT$_t$

Instance: An integer r, a sequence $s_1, \ldots, s_r$ of pairwise disjoint sets of boolean variables, a boolean formula X involving the variables $s_1 \cup \ldots \cup s_r$ which consists of t alternating layers of conjunctions and disjunctions with negations applied only to variables.
Parameter: k
Question: Is it the case that there exists a variable t_1 of s_1 such that for every variable t_2 of s_2 there exists a variable t_3 of s_3 such that ... (alternating quantifiers) such that, when the variables in $t_1, \ldots t_r$ are made true and all other variables are made false, formula X is true?

This problem is AW[t]-complete by [ADF2].

AW[SAT]-Complete

Parameterized QBFSAT

Instance: An integer r, a sequence $s_1, \ldots, s_r$ of pairwise disjoint sets of boolean variables, a boolean formula X involving the variables $s_1 \cup \ldots \cup s_r$, integers $k_1, \ldots, k_r$.
Parameter: r, $k_1, \ldots, k_r$

Question: Is it the case that there exists a size k_1 subset t_1 of s_1 such that for every size k_2 subset t_2 of s_2 there exists a size k_3 subset t_3 of s_3 such that ... (alternating quantifiers) such that, when the variables in $t_1 \cup \ldots \cup t_r$ are made true and all other variables are made false, formula X is true?

Parameterized QCSAT
Instance: An integer r, a sequence $s_1, \ldots, s_r$ of pairwise disjoint sets of boolean variables, a boolean circuit X with the variables $s_1 \cup \ldots \cup s_r$ as inputs, integers $k_1, \ldots, k_r$.
Parameter: r, $k_1, \ldots, k_r$
Question: Is it the case that there exists a size k_1 subset t_1 of s_1 such that for every size k_2 subset t_2 of s_2 there exists a size k_3 subset t_3 of s_3 such that ... (alternating quantifiers) such that, when the inputs in $t_1 \cup \ldots \cup t_r$ are set to 1 and all other inputs are set to 0, circuit X outputs 1?

This problem is AW[SAT]-complete by [ADF2].

AW[SAT]-Hard

Compact DTM Computation 1
Instance: A deterministic Turing machine M operating on tape alphabet Σ, a word $x \in \Sigma^*$, an integer k.
Parameter: k
Question: Does M on input x accept after visiting at most k work tape squares?

This problem is AW[SAT]-hard by a reduction from *Parameterized QBFSAT* $(r, k_1, \ldots, k_r)$ [Ces].

Compact DTM Computation 2
Instance: A deterministic Turing machine M operating on tape alphabet Σ, a word $x \in \Sigma^*$, an integer k.
Parameter: k, $|x|$
Question: Does M on input x accept after visiting at most k work tape squares?

This problem is AW[SAT]-hard by a reduction from *Parameterized QBFSAT* $(r, k_1, \ldots, k_r)$ [Ces].

AW[P]-Complete

Parameterized Monotone QCSAT
Instance: An integer r, a sequence $s_1, \ldots, s_r$ of pairwise disjoint sets of boolean variables, a montone circuit X with the variables $s_1 \cup \ldots \cup s_r$ as inputs, integers $k_1, \ldots, k_r$.

Parameter: r, $k_1, \ldots, k_r$
Question: Is it the case that there exists a size k_1 subset t_1 of s_1 such that for every size k_2 subset t_2 of s_2 there exists a size k_3 subset t_3 of s_3 such that ... (alternating quantifiers) such that, when the inputs in $t_1 \cup \ldots \cup t_r$ are set to 1 and all other inputs are set to 0, circuit X outputs 1?

This problem is AW[P]-complete by a reduction from *Weighted Monotone Circuit Satisfiability* (k) [ADF2].

AW[P]-Hard

Compact NDTM Computation

Instance: A nondeterministic Turing machine M operating on tape alphabet Σ, a word $x \in \Sigma^*$, an integer k.
Parameter: k
Question: Is there an accepting computation of M on input x that visits at most k work tape squares?

This problem is AW[P]-hard by a reduction from *Parameterized QCSAT* $(r, k_1, \ldots, k_r)$ [ADF2].

Compact NDTM Computation

Instance: A nondeterministic Turing machine M operating on tape alphabet Σ, a word $x \in \Sigma^*$, an integer k.
Parameter: k, $|x|$
Question: Is there an accepting computation of M on input x that visits at most k work tape squares?

This problem is AW[P]-hard by a reduction from *Parameterized QCSAT* $(r, k_1, \ldots, k_r)$ [ADF2].

AW[*]=AW[2]-Complete

Short Generalized Geography

Instance: A directed graph $D = (V, A)$, a specified vertex $v_0 \in V$, an integer k.
Parameter: k
Question: Does player one have a winning strategy in k moves for the following game? Players alternately choose a new arc from A. The first arc chosen must have its tail at v_0 and each subsequently chosen arc must have it's tail at the vertex that was the head of the previous arc. The first player unable to choose a new arc loses.

The general version of this problem is PSPACE-complete by a reduction from *QBF* [GJ, GP2]. This problem is AW[*]-complete by a reduction from *Unitary Parameterized* $QBFSAT_t$ (k) [ADF2].

Short Node Kayles
Instance: A graph $G = (V, E)$, an integer k.
Parameter: k
Question: Does player one have a winning k-move strategy in the following game? That is, players pebble a vertex not adjacent to any pebbled vertex. The first player with no play loses. Player one plays first.

The general version of this problem is PSPACE-complete by a reduction from *QBF* [GJ, GP3]. This problem is AW[*]-complete by a reduction from *Unitary Parameterized* $QBFSAT_t$ (k) [ADF2].

In W[1]

Irredundant Set
Instance: A graph $G = (V, E)$, a positive integer k.
Parameter: k
Question: Is there a set $V' \subseteq V$ of cardinality k having the property that each vertex $u \in V'$ has a private neighbor? (A *private neighbor* of a vertex $u \in V'$ is a vertex u' (possibly $u' = u$) with the property that for every vertex $v \in V'$, $u \neq v$, $u' \notin N[v]$.)

Irredundant Set $\in W[1]$.

Proof. Let $G = (V, E)$ be a graph for which we wish to determine if G has a k-element irredundant set. We construct the circuit C corresponding to the following boolean expression E. The variables of E are:

$$\{p[i, x, y] : 1 \leq i \leq k, x, y \in V\} \cup \{c[i, x] : 1 \leq i \leq k, x \in V\}$$

The expression E has the clauses:
(1) $(\neg p[i, x, y] + \neg p[i, x', y'])$ for $1 \leq i \leq k$ and either $x \neq x'$ or $y \neq y'$, $x, x', y, y' \in V$.
(2) $(\neg c[i, x] + \neg c[i, x'])$ for $1 \leq i \leq k$ and $x \neq x'$, $x, x' \in V$.
(3) $(c[i, x] + \neg p[i, x, y])$ for $1 \leq i \leq k$ and $x, y \in V$.
(4) $(\neg c[j, u] + p[i, x, y])$ for $1 \leq i, j \leq k$, $i \neq j$ and $u \in N[y]$.

We argue that the circuit C accepts a weight $2k$ input vector if and only if G has a k-element irredundant set. The clauses of (1) and (2) insure that in any weight $2k$ truth assignment to the variables of E, exactly one of the variables $p[i, x, y]$ is set to *true* for each i, and exactly one of the variables $c[i, x]$ is set to *true* for each i. The meaning we may associate with $p[i, x, y]$ is, "the i^{th} choice of a vertex for the irredundant set is vertex x, and

the private neighbor of x is y." The meaning we may associate with $c[i,x]$ is, "the i^{th} choice of a vertex for the irredundant set is x." The clauses of (3) and (4) enforce the consistency of this interpretation. □

Open Problems

Bounded Hamming Weight Discrete Logarithm

Instance: An n-bit prime, a generator g of F_p^*, an element $a \in F_p^*$, a positive integer k.
Parameter: k
Question: Is there a positive integer x whose binary representation has at most k 1's (that is, x has a Hamming weight of k) such that $a = g$?

Candidate for membership in randomized FPT [FK1,2]. This problem is of practical significance because the use of exponents of fairly small Hamming weight has been suggested in order to speed up cryptosystems based on discrete log (see [FK1,2] and references).

Directed Feedback Vertex Set

Instance: A directed graph $D = (V, A)$, an integer k.
Parameter: k
Question: Is there a set S of k vertices such that each directed cycle of G contains a member of S?

The general version of this problem is NP-complete by a reduction from *Vertex Cover* [GJ, GT8]. This can be solved in $O(n^{k+1})$ by brute force for each fixed k. A related problem *Directed Feedback Arc Set* asks for a set A of at most k arcs such that every directed cycle contains at least one arc from A. These problems can be shown to have the same $\leq_m^s$-degree. The undirected version of this problem is in FPT.

Immersion Order Test

Instance: A graph $G = (V, E)$ and a graph $H = (V', E')$.
Parameter: H
Question: Is $H \leq_i G$ where $\leq_i$ denotes the immersion ordering?

Jump Number

Instance: A poset $P = (P, \leq)$, an integer k.
Parameter: k
Question: Is the jump number of $P \leq k$?

The general version of this problem is NP-complete [See Pulleybank [Pu].]. By El-Zahar and Schmerl [ES], there is an $O(n^{k+1})$ algorithm.

Planar t-Normalized Weighted Satisfiability

Instance: A planar t–normalized formula X and an integer k.
Parameter: k
Question: Does X have a satisfying assignment of weight k?

This question is of some interest for $t = 1$ since it might be a candidate for an intractable problem that is not $W[1]$ hard.

Planar Multiway Cut
Instance: A weighted planar graph $G = (V, E)$ with terminals $\{x_1, \ldots, x_k\}$, an integer k.
Parameter: k
Question: Is there a set of edges of total weight $\leq k'$ whose removal disconnects each terminal from all the others?

The general version of this problem is NP-complete [DJPSY]. Best known complexity is $O((4^k)^k n^{2k-1} \log n)$ by [DJPSY] where it is asked if the probem is FPT.

Polymatroid Recognition
Instance: A k–polymatroid M, an integer k.
Parameter: k
Question: Is M hypergraphic?

See Verigan-Whittle [VW].

Restricted Valence Isomorphism
Instance: Two graphs $G = (V, E)$ and $H = (V', E')$ and an integer k.
Parameter: k
Question: Are G and H isomorphic graphs such that the valencies of the vertices of both G and H are bounded by k?

Luks [Lu] has shown that there is an $O(n^{f(k)})$ algorithm to decide the parameterized version. The question is whether the problem is FPT. If this problem is W-hard then *Graph Isomorphism* is not in P unless the W-hierarchy collapses. The reader should note that this may give an ingress into the Graph Isomorphism problem, in the sense one might be able to demonstrate intractability (i.e. assuming that W[1]≠FPT) of the *un*paramterized version by considering some parameterized version instead. The point is that we have already seen many instances where the "complexity" of the unparameterized version is quite differenrt than tha parameterized one. Consider the fact that the VC-dDimension and Torunament Dominating Set, both of which are LOGNP complete and hence not NP complete unless NP$\subseteq DTIME(n^{\log n})$, yet their parameterized version is $W[1]$ hard.

Short Cheap Tour
Instance: A graph $G = (V, E)$, an edge weighting $w : E \to \mathbf{Z}$, an integer k.
Parameter: k
Question: Is there a tour through at least k nodes of G of cost at most S?

The general version of this problem is NP-complete by a reduction from *Hamiltonian Circuit* [GJ, ND22]. Known to be hard for W[1] if we ask that the tour cost *exactly* S [DF2].

Short Generalized Hex
Instance: A graph $G = (V, E)$ with two distinguished vertices v_1 and v_2, an integer k.
Parameter: k
Question: Does player one have a winning strategy of at most k moves in the following game? Player one plays with white pebbles and player two with black ones. Pebbles are placed on nondistinguished vertices alternately by player one then player two. Player one wins if he can construct a path of white vertices form v_1 to v_2.

The general version of this problem is PSPACE-complete by a reduction from *QBF* [GJ, GP1]. Candidate for AW[*]-completeness [ADF2].

Small Minimum Degree Four Subgraph
Instance: A graph $G = (V, E)$, a positive integer k.
Parameter: k
Question: Is there a subgraph of G of minimum degree at least 4 and of cardinality at most k?

Topological Containment
Instance: A graph $G = (V, E)$ and a graph $H = (V', E')$.
Parameter: H
Question: Is H topologically contained in G?

This can be solved in $O(n^{O(|E'|)})$ time by brute force together with the *k–Disjoint Paths* algorithm of Robertson and Seymour.

Weighted Planar Monotone Boolean Circuit Satisfiability
Instance: A monotone planar decision circuit C, an integer k.
Parameter: k
Question: Does C have a satisfying assignment of weight k?

Candidate for W[SAT]-completeness [ADF2].