Fixed Parameter Tractability and Completeness IV: on Completeness for $W[P]$ and $PSPACE$ analogues

Karl A. Abrahamson,
Department of Computer Science,
East Carolina University,
Greenville
North Carolina, 27858, U.S.A.

Rodney G. Downey [*],
Mathematics Department,
Victoria University,
P.O. Box 600, Wellington, New Zealand.

Michael R. Fellows [†]
Department of Computer Science,
University of Victoria,
Victoria, British Columbia V8W 3P6, Canada

July 7, 2010

[*]Research supported by Victoria University IGC, by the United States / New Zealand Cooperative Science Foundation under grant INT 90-20558, by the University of Victoria, and by the Mathematical Sciences Institute at Cornell University.

[†]Research supported by the National Science and Engineering Research Council of Canada, and the U.S. National Science Foundation under grant MIP-8919312.

Abstract

We describe new results in parameterized complexity theory. In particular, we prove a number of concrete hardness results for $W[P]$, the top level of the hardness hierarchy introduced by Downey and Fellows in a series of earlier papers. We also study the parameterized complexity of analogues of $PSPACE$ via certain natural problems concerning k-move games. Finally, we examine several aspects of the structural complexity of $W[P]$ and related classes. For instance, we show that $W[P]$ can be characterized in terms of the $DTIME(2^{o(n)})$ and NP.

1 Introduction

The theory of NP-completeness provides an excellent vehicle for explaining the apparent asymptotic intractability of many algorithmic problems. Yet while many natural problems do behave intractably *in the limit*, the manner *by which* they arrive at this intractable behavior can vary considerably. The standard NP and other completeness models are often far too coarse to give insight into this variation. To be specific, many natural computational problems take input consisting of two or more parts. The following are some examples of well-known parameterized problems.

Example 1. VERTEX COVER
Instance: A graph $G = (V, E)$.
Parameter: A positive integer k.
Question: Is there a set of vertices $V' \subseteq V$ of cardinality at most k, such that for every edge $uv \in E$, either $u \in V'$ or $v \in V'$?

Example 2. FEEDBACK VERTEX SET
Instance: A graph $G = (V, E)$.
Parameter: A positive integer k.
Question: Is there a set of vertices $V' \subseteq V$ of cardinality at most k such that $G - V'$ is acyclic?

Example 3. GRAPH GENUS
Instance: A graph $G = (V, E)$.
Partameter: A positive integer k.
Question: Can G be embedded on the surface of genus k?

Example 4. MINOR TESTING
Instance: A graph G

Parameter: A graph H
Question: Is $G \geq_m H$?
Example 5. GRAPH LINKING NUMBER
Instance: A graph G.
Parameter: k.
Question: Can G be embedded into 3-space so thst at most k disjoint cycles in G are topologically linked?
Example 6. DOMINATING SET
Instance: A graph $G = (V, E)$.
Parameter: A positive integer k.
Question: Is there a set of vertices $V' \subseteq V$ of cardinality at most k such that for every vertex $u \in V$, there is an edge $uv \in E$ for some vertex $v \in V'$?

With the exception of example 5 for which this question is open, all of the above problems are known to be NP-complete. But what can be said when the parameter k is held fixed? For examples 1-5, there is a constant α such that for every fixed k the problem can be solved in time $O(n^\alpha)$. For examples 3 - 5 we may take $\alpha = 3$ by the deep results of Robertson and Seymour [31] [32]. Example 6 illustrates the contrasting situation where for fixed values of k we seem to be able to do no better than a brute force examination of all possible solutions and thus the best known algorithm is $O(n^k)$. The contrast is that for problems such as DOMINATING SET above, INDEPENDENT SET and many others, for a fixed k there only seem algorithms running in time $O(n^{g(k)})$ *with* $g(k) \to \infty$. We are thus concerned with an issue that is very much akin to P *versus* NP. The previous papers of this series [10] [11] [12] [13], established a framework with which to address the apparent fixed-parameter intractability of problems such as example 6. We feel that our framework provides an important contribution to the analysis of complexity of combinatorial problems for the following reasons.

(i) Distinct from most other notions and classes inroduced since the original clarification of NP and $PSPACE$ completeness, our framework is applicable to a *wide* class of *practical* problems.

(ii) Our framework provides a refined measure to that helps to explain the apparent diversity of *actual* behavior of many hard problems, as well as having numerous other applications.

This theory has found applications to various concrete problem domains such as cryptography (Fellows-Koblitz [21]), computational biology (Fellows-Hallett-Wareham [20]), computational learning theory (Downey-Fellows [15],

Downey-Evans-Fellows [9]), and many others. The theory has ties with such classical notions as advice classes (Cai *et. al.* [6]) and some of the basic structural questions raised by, for instance, the various reducibilities, seem currently beyond our capabilities. (Even ones which are easy for classical reducibilities. See e.g. Downey-Fellows [14]). The issues raised in the study of parameterized complexity seem to have very wide ranging theoretical and practical interest.

In [10], [11] a hierarchy, the W-hierarchy,

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

is defined and studied, where the base class FPT denotes the fixed-parameter tractable problems. This hierarchy is based on the logical depth needed to describe the problems in terms of circuits. (The formal definitions we shall need are given in §2.) It is shown, for example, that INDEPENDENT SET (equivalently, CLIQUE) is complete for $W[1]$ ([12]), and that DOMINATING SET is complete for $W[2]$ ([11]). (For a compendium of the many known completeness and hardness results see [11].)

We conjecture that the above hierarchy of parameterized problem classes is proper. If $P = NP$ then $FPT = W[P]$, and conversely, if $FPT = W[P]$ then, as we will prove, a *quantitative* version of the $P \neq NP$ conjecture fails.

Since for each fixed parameter value, each of the problems in the class $W[P]$ is solvable in polynomial time, this theory addresses in some sense (i.e., with parameters fixed) complexity issues *inside of* P. Alternatively, one may view the larger issue as regarding limited amounts of nondeterminism. For related studies addressing these issues see for instance, [3], [4], [6], [29], and [30])

In the present paper, we will first concentrate upon establishing some concrete $\cup_{i\in\omega} W[i]$-hardness results. In particular, we will concentrate upon $W[P]$, the top of the W-hierarchy, as well as a new class we call $W[SAT]$ which is again apparently distinct from $W[P]$, and contains a number of important combinatorial problems such as the problem of deciding WEIGHTED SATISFIABILITY: does a given formula φ of propositional logic have a satisfying assignment with exactly k literals true. These sections will complement the earlier papers by Downey and Fellows which established hardness results for lower levels such as $W[1]$ and $W[2]$. As an illustration, we shall establish that the following problems (and many others) are complete for $W[P]$.

DEGREE 3 SUBGRAPH ANNIHILATOR
Instance: A graph G.
Parameter: k
Question: Is there a set V' of at most k vertices of G, such that $G - V'$ has no minimum degree subgraph?
WEIGHTED PLANAR CIRCUIT SATISFIABILITY
Instance: A planar decision circuit C.
Parameter: k.
Question: Does C have a *weight* k satisfying assignment? (A weight k satisfying assignment is one with exactly k variables set to be true.)
k-LINEAR INEQUALITIES
Instance: A system of linear inequalities.
Parameter: k.
Question: Can the system be made consistent over the rationals by deleting at most k of the inequalities?

The second goal of the present paper is to introduce and study parameterized analogues of problems complete for $PSPACE$. This turns out to be of some interest since we are naturally lead to the analysis of the computational complexity of k-move games, such as ALTERNATING HITTING SET.

Finally in the last section we shall make some contributions to the study of the structural complexity of the notions we have introduced. For instance, as we mentioned earlier, we shall prove that if the W-hierarchy collapses, then a qualitiative version of $P = NP$ holds. This is expressed in terms of a notion we call *near polynomial time* which is along the lines of a $DTIME(2^{o(n)})$ algorithm for SATISFIABILITY. We also show that certain quantative classical complexity- theoretical collapses occur if $FPT = W[P]$.

The paper is self-contained and can be read independently of the others in the series. Preliminary versions of some of the results appeared in the extended abstract [1]. We thank Liming Cai for helpful discussions concerning aspects of this work.

2 Preliminaries

A *parameterized problem* is a set $L \subseteq \Sigma^* \times \Sigma^*$. Typically, the second component represents a parameter $k \in N$, and for our purposes can be thought of

as being presented in unary. For $k \in \Sigma^*$ we write $L_k = \{y|(y,k) \in L\}$. We refer to L_k as the k-th *slice* of L.

Consideration of examples 1-5 of §1 leads to three flavours of tractability and reduction.

Definition of Fixed-Parameter Tractability. We say that a parameterized problem L is

(1) *nonuniformly fixed-parameter tractable* if there is a constant α and a sequence of algorithms Φ_x such that, for each $x \in N$, Φ_x computes L_x in time $O(n^\alpha)$;

(2) *uniformly fixed-parameter tractable* if there is a constant α and a *single* algorithm Φ such that Φ decides if $(x,k) \in L$ in time $f(k)|x|^\alpha$ where $f : N \to N$ is an arbitrary function;

(3) *strongly uniformly fixed-parameter tractable* if L is uniformly fixed-parameter tractable with the function f recursive.

Example 1 is strongly uniformly f.p.tractable (as are most examples of f. p. tractability obtained without essential use of the Graph Minor Theorem). Example 2 can be shown to be strongly uniformly FPT by the methods of [23] (the Graph Minor Theorem alone gives only give nonuniform tractability). Example 3 can be shown to be uniformly FPT by the method of [22] (since the technique of [23] is not presently known to apply, we do not know a strongly uniform algorithm). Example 5 is at present only known to be nonuniformly FPT. If $P = NP$ then example 6 is also FPT. The following is the basic definition needed for a completeness programme to help explain the apparent fixed parameter *intractability* of such problems.

Definition (Uniform Reducibility). Let A, B be parameterized problems. We say that A is *uniformly f.p.-reducible* to B if there is an oracle algorithm Φ, a constant α, and an arbitrary function $f : N \to N$ such that

(a) the running time of $\Phi(B; \langle x,k \rangle)$ is at most $f(k)|x|^\alpha$,

(b) on input $\langle x,k \rangle$, Φ only asks oracle questions of $B^{(f(k))}$ where

$$B^{(f(k))} = \bigcup_{j \leq f(k)} B_j = \{\langle x,j \rangle : \langle x,j \rangle \in B\}$$

(c) $\Phi(B) = A$.

If A is uniformly f.p.-reducible to B, write $A \leq_T^u B$. We may say that $A \leq_T^u B$ *via* f. If the reduction is many:1 (an *m-reduction*), write $A \leq_m^u B$.

Working Definition of Parameterized Reducibility. As with classical NP-completeness results, most of the reductions we construct will be *m*-

reductions, and in fact will be of the form

$$\langle x,k\rangle \in A \text{ iff } \langle g(x,k),h(k)\rangle \in B.$$

Indeed, in all cases in the present paper, g will be P-time of low degree and h will be recursive. Such reducibilities are special cases of the *strong uniform reducibility* described in the next definition.

Definition. (i) We say that A is *strongly uniformly f.p.-reducible* to B if $A \leq_T^u B$ via f where f is recursive. We write $A \leq_T^s B$ in this case.
(ii) We say that A is *nonuniformly f.p.-reducible* to B if there exists a constant α, a function $f : N \to N$, and a collection of procedures $\{\Phi_k : k \in N\}$ such that $\Phi_k(B^{(f(k))}) = A_k$ for $k \in N$, and the running time of Φ_k is $f(k)|x|^\alpha$. Here we write $A \leq_T^n B$.

Note that the above are good definitions since whenever $A < B$ with $<$ any of the reducibilities, if B is FPT so too is A. We will write $FPT(\leq)$ for the FPT class corresponding to the reducibility $\leq$.

(2.1) Theorem. *[14] The ordering $\leq_T^s$ partitions $FPT(\leq_T^u)$ into infinitely many classes. Similarly $\leq_T^u$ partitions $FPT(\leq_T^n)$ into infinitely many classes.*

Fix attention on any of the above reducibilities, and call it *fp-reducibility*. We consider circuits (termed *mixed type*) in which some gates have bounded fan-in (*small gates*) and some have unrestricted fan-in (*large gates*).

Definition. The *depth* $d(C)$ of a circuit C is the maximum number of gates (small or large), on an input-output path in C. The *weft* $w(C)$ 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 exists a constant h such that $\forall C \in F$, $d(C) \leq h$. We say that F has *bounded weft* if there exists a constant t such that $\forall C \in F$, $w(C) \leq t$. F is a *decision circuit family* if each circuit has a single output. A decision circuit C *accepts* an input vector x if the single output gate has value 1 on input x. The *weight* of a boolean vector x is the number of 1's in the vector.

Definition of the W-hierarchy. 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 $L_F = \{(C,k) : C \in F$ and C accepts an input vector of weight $k\}$.
(i) A parameterized problem L belongs to $W[t]$ if L reduces to the parameterized decision circuit problem $L_{F(t,h)}$ for the family $F(t,h)$ of mixed type

decision circuits of weft at most t, and depth at most h, for some constant h.
(ii) A parameterized problem L belongs to $W[P]$ if L reduces to the circuit problem L_F where F is the set of all circuits (no restrictions).
(iii) A prameterized problem belongs to $W[SAT]$ if L reduces to the circuit problem L_F where F is the set of all *boolean* circuits. (I.e all circuits with all gates having fanout 1.)

The above leads to an interesting hierarchy of parameterized problem classes

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

If $P = NP$ then the hierarchy collapses. As we mentioned in the introduction, we conjecture that each of the containments is proper.
Current Working Hypothesis: Our current measure of fixed parameter *intractability* is $W[1]$-hardness. [As we see in Theorem (2.2) below this means that L is fixed parameter intractable if L is a parameterized problem that is as hard as deciding if a 3CNF proposition has a weight k satisfying assignment.]
Motivation: The above hierarchy may seem slightly strange at first but we mention the following result which establishes why the notion of weft is natural. We say that a formula φ of propositional logic is in q-CNF or q-PoS (Product of Sums), if φ is the conjunction of clauses of bounded size q. We drop the q and write CNF or PoS in the situation that there is no restriction on clause size and we can similarly generate PoSoPoSo...oP(S). With r alternations we call such formulae r-NORMALIZED formulae. With this definition we can generate the problem of WEIGHTED r-NORMALIZED SATISFIABILITY which asks, for a parameter k, if a given r-normalized formula has a weight k satisfying assignment. The following result is an amalgam of work of Downey and Fellows [10], [11], and [12]. The proof is not straightforward. (The full proof of (iii) is given here for completeness.)
(2.2) Theorem. (Downey and Fellows) *(i) For $q \geq 2$* WEIGHTED q-CNF SATISFIABILITY *is $W[1]$-complete.*
(ii) For all $r \geq 2$, WEIGHTED r-NORMALIZED SATISFIABILITY *is $W[r]$-complete.*
(iii) WEIGHTED SATISFIABILITY *is $W[SAT]$-complete.*

3 Some $W[P]$-Completeness Results.

We begin this section with the following a series of basic $W[P]$ completeness results. The following problem provides a useful variation on WEIGHTED CIRCUIT SATISFIABILITY for basing combinatorial reductions.
SHORT CIRCUIT SATISFIABILITY
Instance: A decision circuit C with at most n gates and $\leq k \log n$ inputs.
Parameter: k
Question: Is there any setting of the inputs that causes C to output 1?
(3.1) Lemma. SHORT CIRCUIT SATISFIABILITY *is* $W[P]$*-Complete.*
Proof. To see that it is $W[P]$-hard, take an instance C of WEIGHTED CIRCUIT SATISFIABILITY, with parameter k and inputs $x_1, ..., x_n$. Let $z_1, ..., z_{k \log n}$ be new variables. Using lexicographic order in polynomial time (independent of k) we can have a surjection from $Z = \{z_1, ..., z_{k \log n}\}$ to the (the characteristic function of the) k-element subsets of $X = \{x_1, ..., x_n\}$. Representing this as a circuit with inputs Z and outputs X we can put this circuit on top of C to for C' so that C accepts a weight k input iff C' accepts some input.

That $W[P]$ contains SHORT CIRCUIT SATISFIABILITY is equally easy, just use the polynomial embedding of the $k \log n$ into the $k+1$-element subsets of n for n sufficiently large. □

For the following result we shall employ ideas from Abrahamson *et. al.* [2]. Let φ be a formula of propositional logic (or a circuit). Suppose we wish to extend a partial truth assignment t of a (circuit or) formula φ to a satisfying assignment t'. If there is a clause, say, all but one of whose literals are forced to be false by t then t implicitly causes the remaining literal to be made *true*. Repeat this process until we can go no further. If this process generates t' a satisfying assignment of φ all of whose literals are specified we say that t causes φ to *unravel*. Following [2], we have the following problem definitions.
SHORT SATISFIABILITY
Input: A formula φ of n variables and a list of at most $k \log n$ variables of φ.
Parameter: k.
Question: Is ther any setting of the distinguished variables that causes φ to unravel?
k-INDUCED SATISFIABILITY
Input: A formula φ.

Paramater: k.
Question: Is there a set of k variables, and a truth table assigment to those variables that causes φ to unravel?
(3.2) Theorem. *The following are $W[P]$-Complete:*
(i) SHORT SATISFIABILITY.
(ii) k-INDUCED SATISFIABILITY.

Proof. (i) We reduce from SHORT CIRCUIT SATISFIABILITY via (3.1). Let $x_1, ..., x_m, C$ be an instance of SHORT CIRCUIT SATISFIABILITY with parameter k and $m \leq k \log n$. These variables will be the distinguished variables for the input $\varphi = \varphi(C)$. We introduce new variables and defing φ by locally replacing gates of C as follows. We replace a *not* gate $y = \overline{x}$ by the formula $(x \vee y) \wedge (\overline{x} \vee \overline{y})$ and an *and* gate $z = x \wedge y$ by $(x \vee \overline{z}) \wedge (y \vee \overline{z}) \wedge (\overline{x} \vee \overline{y} \vee z)$. The formula generated by the circuit is at most quadraticly larger than the circuit, and clearly a setting of $x_1, ..., x_m$ satisfies C iff it unravels φ.
(ii) The proof is delayed till Theorem (3.7). □

The basis for many of our reductions is the $W[P]$-completeness of WEIGHTED MONOTONE CIRCUIT SATISFIABILITY, which is the restriction of the problem WEIGHTED CIRCUIT SATISFIABILITY to circuits with no inverters. This result was first announced in Downey *et. al.* [17]. In [17], Downey *et. al* analysed the parameterized tractability of logical problems stemming from logic and linguistics. The problem of relevance to the present paper is the following.
MINIMUM AXIOM SET
Input: A finite set S of "sentences," an "implication relation" R consisting of pairs (A, t) where $A \subseteq S$ and $t \in S$, and a positive 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 $(U, t) \in R$, then $S_n = S$?

This problem is, of course, the parameterized version of a well known NP complete problem from Garey and Johnson [25].
(3.3) Theorem. (Downey, Fellows, Kapron, Hallett, and Wareham [17]) MINIMUM AXIOM SET *is $W[P]$-Complete.*

The proof consists of the following two lemmata. We prove this result in full detail for both the sake of completeness and because the proof is only

sketched in [17].

(3.4) Lemma. MINIMUM AXIOM SET *belongs to monotone* $W[P]$.

Proof. We describe a parameterized polynomial-time transformation of an instance (S, R) of MINIMUM AXIOM SET to a circuit C that accepts a weight k vector if and only if (S, R) has an axiom set of size k. For convenience of description, we view C as a directed acyclic graph for which (1) each vertex is assigned a logic function in the set $\{\wedge, \vee, 1\}$, where 1 denotes the identity function, and (2) some (appropriate) vertices have been designated as inputs.

Suppose $|S| = n$. The vertex set of C is $V(C) = V_1 \cup V_2$ where

$$V_1 = \{t[u, i] : \ u \in S, \ 0 \leq i \leq n\}$$

Each vertex of V_1 is assigned the identity logic function. The inputs to C are the vertices $\{t[u, 0] : u \in S\}$. We may think of each vertex of V_1 in the circuit C are representing a boolean variable, the meaning of which is, "statement u is true at time i." Thus a weight k input to C represents k statements (the axioms) being true at time 0.

The vertices of V_2 are the union of n^2 sets of vertices

$$V_2 = \bigcup \{V[u, i] : \ u \in S, \ 1 \leq i \leq n\}$$

where the vertices of $V[u, i]$ implement a monotone circuit $C(u, i)$ that computes the value of $t[u, i]$ from the value of variables at the $i - 1$ level of V_1, in accordance with the definition of the sets S_i in the description of the MINIMUM AXIOM SET problem. We may describe $C(u, i)$ by the sum-of-products expression

$$C(u, i) = t[u, i - 1] + \Sigma_{(A,u) \in R} \left(\prod_{v \in A} t[v, i - 1] \right)$$

The correctness of this construction is straightforward, noting that by time n, all of the statements deducible from the axioms will have been deduced. □

(3.5) Lemma. MINIMUM AXIOM SET *is hard for* $W[P]$.

Proof. Let C be a decision circuit for which we wish to determine whether there is a Boolean vector of weight k accepted by C. We describe how C can be transformed in parameterized polynomial time into an instance (S, R) of

the *Minimum Axiom Set* problem that has an axiom set of size $2k$ if and only if C accepts a weight k input vector.

As in the proof of Lemma (3.5), we view the circuit C as a directed acyclic graph $C = (V, A)$. We may assume that the circuit C is *leveled* in the sense that the vertices of V are partitioned into m sets $V[1], ..., V[t]$ ("levels"), with the inputs constituting level 1, and such that for every arc $uv \in A$, if u is in level i then v is in level $i + 1$. Since C is a general circuit, each vertex of $V - V[1]$ is assigned a logic function from the set $\{\wedge, \vee, \neg\}$. Thus $V - V[1]$ can be expressed as the disjoint union

$$V - V[1] = V_{\wedge} \cup V_{\vee} \cup V_{\neg}$$

We assume without loss of generality that $\wedge$ and $\vee$ vertices have in-degree two and that $\neg$ vertices have in-degree one. We will assume that C has n inputs $V[1] = \{u_0, u_1, ..., u_{n-1}\}$. Let z denote the output vertex of C.

The set of "statements" S for the instance (S, R) of *Minimum Axiom Set* is the union of three sets, $S = S[1] \cup S[2] \cup S[3]$. Similarly, we describe the deductive structure R as the union of several parts, $R = R[1] \cup R[2] \cup R[3] \cup R[4] \cup R[5]$. The deductive structure $R[1]$ on $S[1]$ has the primary role of simulating the logic gates of the circuit C. The deductive structure $R[2]$ on $S[1]$ and $S[2]$ handles the inputs to the simulation of C. The role of $R[3]$ and $R[4]$ is to serve an enforcement mechanism for the input simulation. The role of $R[5]$ is in handling the output of the simulation of C.

$$S[1] = \{q[u, i] : \ u \in V, \ i \in \{0, 1\}\}$$

The simulation of C provided by the deductive structure on $S[1]$ is straightforward. Note that each vertex u of C is represented by two "statements" $q[u, 0]$ and $q[u, 1]$ in $S[1]$. The deductive structure (S, R) simulates C in the following general way. First, the structure provided by $S[2]$ enforces (in a manner explained below) that "at time 1" exactly one of $q[u, 0]$ and $q[u, 1]$ has been deduced for each input vertex $u \in V_0$ (that is, will belong to the set S_1 in the definition of the MINIMUM AXIOM SET problem), for any $2k$ element axiom set with any hope of success (let us call this "meaningful input" for the moment). Inductively, the simulation of C insures that for meaningful input, for each vertex $q \in V$, and for $i = 1, ..., t$, exactly one of $q[u, 0]$ and $q[u, 1]$ will belong to S_i.

In describing the simulation, we interpret this in the following way: if $q[u,0] \in S_t$ then the vertex (logic gate) q of C has logic value 0 at time t, and if $q[u,1] \in S_t$ then q has logic value 1 at time t. It may eventually transpire (at time t, since the circuit is leveled) that for the output vertex z of C, the statement $q[z,1]$ is deduced. The structure (S,R) is such that if $q[z,1]$ is deduced (at time t) then "everything" in S can be deduced at time $t+2$. The details of the circuit simulation are as follows.

For $u \in (V_\wedge \cup V_\vee)$ let u' and u'' denote the input vertices to u, that is, the two vertices for which $u'u$ and $u''u$ are in the set of arcs A describing the circuit. Similarly, for $u \in V_\neg$ let u' denote the single input vertex to u. The implications of $R[1]$ provide the circuit simulation. We have $R[1] = R_\wedge \cup R_\vee \cup R_\neg$ where

$$R_\wedge = \bigcup_{u \in V_\wedge} \{(\{q[u',1], q[u'',1]\}, q[u,1]), (\{q[u',0]\}, q[u,0]), (\{q[u'',0]\}, q[u,0])\}$$

$$R_\vee = \bigcup_{u \in V_\vee} \{(\{q[u',0], q[u'',0]\}, q[u,0]), (\{q[u',1]\}, q[u,1]), (\{q[u'',1]\}, q[u,1])\}$$

$$R_\neg = \bigcup_{u \in V_\neg} \{(\{q[u',1]\}, q[u,0]), (\{q[u',0]\}, q[u,1]), \}$$

We next describe the structure which simulates the input to the circuit. This construction is based on the ideas of Theorem 2.1 of [10]. We have $S[2] = S[2,1] \cup S[2,2] \cup S[2,3] \cup S[2,4] \cup S[2,5]$ where

$$S[2,1] = \{a[r,s,i] :\ 0 \le r \le k-1,\ 0 \le s \le n-1,\ 1 \le i \le 2\}$$

$$S[2,2] = \{b[r,s,t,i] :\ 0 \le r \le k-1,\ 0 \le s \le n-1,\ 1 \le t \le n-k+1,\ 1 \le i \le 2\}$$

$$S[2,3] = \{a'[r,u,i] :\ 0 \le r \le k-1,\ 1 \le u \le 2k+1,\ 1 \le i \le 2\}$$

$$S[2,4] = \{b'[r,u,i] :\ 0 \le r \le k-1,\ 1 \le u \le 2k+1,\ 1 \le i \le 2\}$$

$$S[2,5] = \{d[r,s,i] :\ 0 \le r \le k-1,\ 0 \le s \le n-1,\ 1 \le i \le 2\}$$

Note that the set of statements $S[2]$ can be partitioned into two sets $S[2] = S'[2] \cup S''[2]$ according to the value of the last index of the elements. That is, let $S'[2]$ be those elements of $S[2]$ with last index $i = 1$ and let $S''[2]$ be those elements of $S[2]$ with last index $i = 2$. For $x' \in S'[2]$ let x'' be the element of $S''[2]$ that corresponds to x' by changing the last index from $i = 1$ to $i = 2$. Similarly we define the notation $S'[2,j]$ for $j = 1, ..., 5$.

We can now describe the second set of implications

$$\begin{aligned} R[2] &= \{(\{a[r,i,1] : 0 \le r \le k-1\}, q[u_i,1]) : 0 \le i \le n-1\} \\ &\quad \cup \{(\{b[r,s,t,1]\}, q[u_i,0]) : \ s < i < s+t, \ 0 \le i \le n-1\} \end{aligned}$$

We next introduce as an intermediate descriptive device a graph $G = (V, E)$ (the *domination graph*) on the vertex set $V = S'[2]$. For convenience, we introduce notation for the following sets of vertices in this graph.

$$A(r) = \{a[r,s,1] : 0 \le s \le n-1\}$$

$$B(r) = \{b[r,s,t,1] : 0 \le s \le n-1, 1 \le t \le n-k+1\}$$

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

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.

$$E_1 = \{a[r,s,1]a[r,s',1] : s \ne s'\}$$

$$E_2 = \{b[r,s,t,1]b[r,s,t',1] : t \ne t'\}$$

$$E_3 = \{a[r,s,1]b[r,s',t,1] : s \ne s'\}$$

$$E_4 = \{b[r,s,t,1]d[r,s',1] : s' \ne s+t \pmod n\}$$

$$E_5 = \{a[r,s,1]a'[r,u,1]\}$$

$$E_6 = \{b[r,s,t,1]b'[r,u,1]\}$$

$$E_7 = \{d[r,s,1]a[r',s,1] : r' = r+1 \pmod n\}$$

We use this graph in describing the third set of implications as follows. For a vertex $v \in V$, the *closed neighborhood* $N[v]$ *of* v is defined to be $N[v] = \{w : vw \in E\} \cup \{v\}$.

$$R[3] = \{(N[x'], x'') : x' \in V = S'[2]\}$$

The third set of statements is

$$S[3] = \{e[i] : 1 \le i \le 2k+1\}$$

The final sets of implications are

$$R[4] = \{(S''[2] \cup \{q[z,1]\}, e[i]) : 1 \le i \le 2k+1\}$$

$$R[5] = \{(S[3], s) : s \in S[1] \cup S[2]\}$$

This completes the description of the instance (S, R) of MINIMUM AXIOM SET. We argue the correctness of the reduction as follows.

We first describe an *interpretation* of a set U of $2k$ vertices in the domination graph, as this plays a key role in the argument. We interpret $a[r, s, 1] \in U$ as the directive, "set the input u_s of C to 1." We interpret $b[r, s, t, 1] \in U$ as the directive, "set the inputs u_i of C to 0, for all i, $s < i < s + t$." If these directives are consistent for the vertices in a set U, call U a *consistent* set of vertices in the domination graph. Say that U is *total* if the directives corresponding to the vertices in U assign each input to C a value. If U is consistent and total, then the *interpretation* τ_U of U is the input to C corresponding to U according to the directives.

Claim 1. A dominating set D of $2k$ vertices in the domination graph G is is consistent and total and the corresponding Boolean input vector τ_D to the circuit C has weight k.

Proof of Claim 1.

Suppose D is a dominating set of $2k$ vertices in G. The closed neighborhoods of the $2k$ vertices $a'[0, 1, 1], ..., a'[k - 1, 1, 1], b'[0, 1, 1], ..., b'[k - 1, 1, 1]$ 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 $a'[r, u, 1] \in D$ then necessarily $a'[r, u', 1] \in D$ 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_3, E_4 and E_7. The vertices of D in $S'[2, 1]$ represent the inputs set to 1 in a weight k input vector for the circuit C, and the vertices of D in $S'[2, 2]$ represent "intervals" of inputs set to 0 in a weight k input vector for C. Since there are k inputs to be set to 1 (in a weight k vector), there are, considering the indices of the variables mod n, also k intervals of inputs to be set to 0.

The edges of E_3, E_4 and E_7 enforce that the $2k$ vertices in D must represent such a choice consistently. To see how this enforcement works, suppose $a[3, 4, 1] \in D$. This represents that the third of k distinct choices of inputs to be given the value 1 is the input u_4. The edges of E_3 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 an input to receive the value 1, and thus the vertex represents a range of inputs to receive the value 0. The edges of E_4 and E_7 enforce that the index t of the vertex of D in the subset $B(3,4)$ represents the "distance" to the next input to receive the value 1, as it is represented by the unique vertex of D in the set $A(4)$.

In this way, a dominating set D of size $2k$ is forced to be both consistent and total, and we may note also that τ_D necessarily has weight k. □

Claim 2. If C accepts a weight k input vector, then (S, R) is a yes-instance of MINIMUM AXIOM SET.

Proof of Claim 2. Let τ denote an input Boolean vector of weight k accepted by C, that assigns the inputs $V_\tau = \{u_{i_0}, u_{i_1}, \ldots, u_{i_{k-1}}\} \subseteq V[1]$ the value 1. 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

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

is a dominating set in the domination graph G and that $\tau_D = \tau$.

We take the initial set of axioms $S_0 = D$. We will argue that $S_{t+2} = S$. Since D is a dominating set in the graph on $S'[2]$, by the implications in $R[3]$ (based on closed neighborhoods in the domination graph) we may deduce that $S''[2] \subseteq S_1$.

The implications in $R[2]$ yield that

$$S[1] \cap S_1 = \{q[u, 1] : u \in V_\tau\} \cup \{q[u, 0] : u \in V[1] - V_\tau\}$$

consistent with our description of the circuit simulation (at time 1).

By induction on i, noting the form of the implications in $R[1]$, it is straightforward that the circuit simulation is correct; that is, for $i = 2, ..., t$ we have

$$\begin{aligned} S[1] \cap S_i \;\; = \;\; & \{q[u, 1] : \; u \text{ evaluates to 1 in } C(\tau) \; \} \\ & \cup \{q[u, 0] : \; u \text{ evaluates to 0 in } C(\tau) \; \} \end{aligned}$$

Since C accepts τ, $q[z, 1] \in S_t$, and consequently by the implications in $R[4]$, $S[3] \subset S_{t+1}$. By the implications in $R[5]$ this yields $S \subseteq S_{t+2}$. □

Claim 3. If $(S, R, 2k)$ is a yes-instance of MINIMUM AXIOM SET then C accepts a weight k input vector.

Proof of Claim 3. Let S_0 be the set of $2k$ axioms. If S_0 does not contain $2k$ statements in $S'[2]$ then by Claim 1 S_0 is not a dominating set in the

domination graph, and consequently there is at least one statement $s \in S''[2]$ that can only be deduced if every statement in $S[3]$ is first deduced (or is an axiom). Not every statement in $S[3]$ can be an axiom (since the cardinality of $S[3]$ is $2k+1$), so there must be some statement $s' \in S[3]$ that is properly deduced, necessarily by an implication in $R[4]$, prior to the deduction of s. An examination of $R[4]$ shows that s must be deduced prior to s', a contradiction. Thus $S_0 \subseteq S'[2]$, and furthermore, S_0 must be a dominating set in G.

By Claim 1, S_0 must be consistent and total, and we may therefore consider the interpretation τ_{S_0}, which we argue is accepted by C. The only possibility for statements in $S'[2] - S_0$ to be deduced is by the implications in $R[5]$, and these can only be applied if $q[z,1]$ has been deduced. By the inductive argument for the correctness of the circuit simulation, this can only happen if C accepts the weight k input vector τ_{S_0}. □

That completes the proof of Lemma (3.5). □

(3.6) Corollary. WEIGHTED MONOTONE CIRCUIT SATISFIABILITY *is* $W[P]$*-complete. That is,* MONOTONE $W[P] = W[P]$.

Proof. By Lemma (3.4) MINIMUM AXIOM SET is a special case of WEIGHTED MONOTONE CIRCUIT SATISFIABILITY. □

The basic results above form a basis for many of the reductions for various concrete problems. The next theorem gives some illustrations. We will need the following definitions:

CHAIN REACTION CLOSURE
Instance: A directed Graph D.
Parameter: k. *Question:* Does there exist a set V' of k vertices of D whose chain reaction closure is D. Here the 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 $x \in S$.

t-THRESHOLD STARTING SET
Instance: A digraph D.
Parameter: k.
Question: Is there a set of k vertices of D with the property that it will pebble D under the rule that a vertex can be pebbled iff at least t incoming vertices are pebbled?

(3.7) Theorem. *The following problems are* $W[P]$*-complete.*
(i) WEIGHTED PLANAR CIRCUIT SATISFIABILITY
(ii) DEGREE 3 SUBGRAPH ANNIHILATOR

(iii) CHAIN REACTION CLOSURE
(iv) t-THRESHOLD STARTING SET
(v) k-INDUCED 3CNF SATISFIABILITY
(vi) k-LINEAR INEQUALITIES

Proof. (i) We reduce WEIGHTED CIRCUIT SATISFIABILITY to the planar version. This involves no more than the use of a crossover gadget along the lines of Lichtenstein [28], or Garey and Johnson [24]. Take an instance φ of WEIGHTED CIRCUIT SATISFIABILITY with paramater k. Suppose a wire from x (a variable or gate) is crossing one from y. We replace this crossing by the gadget of diagram 1. On this diagram we have listed the values at each gate as a triple corresponding to the settings of x and y. Note that we need only give a triple by symmetry. Since p agrees with y and q agrees with x we have achieved the desired crossing.
(ii) We reduce from WEIGHTED MONOTONE CIRCUIT SATISFIABILITY. Let C be an instance of WEIGHTED MONOTONE CIRCUIT SATISFIABILITY with parameter k. We can regard all gates to have only two inputs with only quadratic increase in the size of the circuit. We shall use local replacement. We replace the *and* and *or* gates by the gadgets given in diagram 2. Formally, we can define the *and* gadget $AND(x, y, z)$ for a gate $x \wedge y$ with output z via the vertices
$v(x, j)$ and $v(y, j)$ for $1 \leq j \leq 7$ together with vertex $q(x, y)$ and "pendant vertices" $l(x, y, p)$ for $p = 1, 2, 3, 4$.
The edges are for $r \in \{x, y\}$,
$rv(r, 1)$,
$v(r, 1)v(r, 2),\ v(r, 1)v(r, 3),\ v(r, 2)v(r, 4),\ v(r, 3)v(r, 5),\ v(r, 4)v(r, 5)$,
(These are the "pentagonal" edges of the gadget.)
$v(r, 2)v(r, 6),\ v(r, 4)v(r, 6),\ v(r, 6)v(r, 7),\ v(r, 3)v(r, 7),\ v(r, 5)v(r, 7)$,
(These are the edges connected to the pentagon.)
$v(r, 6)q(x, y),\ v(r, 7)q(x, y),\ q(x, y)l(x, y, 1)$,
(These are the connecting edges.)
$l(x, y, 1)l(x, y, 2),\ l(x, y, 1)l(x, y, 3),\ l(x, y, 2)l(x, y, 3),\ l(x, y, 2)l(x, y, 4)$,
and $l(x, y, 3)l(x, y, 4)$
(These are the edges of the "pendant diamond").

The vertices for the *or* gadget $OR(x, y, z)$ are $w(x, j)$, $w(y, j)$ for $1 \leq j \leq 7$ and 3 additional vertices $m(x)$, $m(y)$, $n(x, y)$ and $g(x, y, p)$ for $p = 1, 2, 3, 4$. The vertices $w(r, j)$ are connected among themselves exactly as we

did for the $v(r,j)$ above, as are the pendant $g(x,y,p)$. We also have edges $w(r,6)m(r)$, $w(r,7)m(r)$, $m(r)n(x,y)$, and $n(x,y)z$.

Additionally, we shall need another sort of gadget, a *fanout* gadget which is used to cope with the fanout occurring at gates and in the construction. This gadget is also indicated in diagram 2 for fanouts 3 and 4.

Let the variables of C be $x[1],...,x[n]$. For the construction of the graph $G(C)$, we first replace all the gates of C and their fanouts by the gadgets of diagram 1. Call the result $I(C)$. Using $I(G)$, we make another intermediate graph $I'(G)$ as follows. For each $x[i] \in \{x[1],...,x[n]\}$, create a subgraph consisting of the (new) vertices $v(i,j) : j = 1,...,4$ and edges $v(i,1)v(i,2)$, $v(i,1)v(i,3)$, $v(i,2)v(i,3)$, $v(i,2)v(i,4)$, and $v(i,3),v(i,4)$. form $2k+1$ copies of the $I(C)$ except that in place of the $k+1$ versions of the input variable $x[i]$ we connect all relevant gates to the *single* vertex $v(i,4)$. (Thus if in the original circuit the variable $x[i]$ had fanout m then in the graph $I'(C)$ $v(i,4)$ will have degree $m(k+1)+2$.) Finally we form the final graph $I''(C)$ by taking the vertex $u(i)$ representing the output of the i'th version of $I(C)$ used in $I'(C)$, and for each $j = 1,...,n$, join $o(i)$ to $v(j,1)$ *via* one of the outputs u_j of a fanout gadget of fanout n. This concludes the construction.

To see that the construction succeeds, we show that C has a a weight k accepting input iff $I''(C)$ has no k element degree 3 subgraph annihilator. First suppose that C has a weight k accepting input. Let $x[i_1],...,x[i_k]$ be the true variables in some satisfying assignment. Remove the vertices $v[q,4]$ for $q \in \{i_1,...,i_k\}$ from $I''(C)$. We claim that the result has no min degree 3 subgraph. We shall describe a process that removes edges that are useless to a min degree 3 subgraph and eventually kills all of $I''(C)$. The reader should note the following guiding principle to our proof. Given any graph G the following simple process determines if G has a min degree 3 subgraph.

1. Repeat until there are no degree ≤ 2 vertices:
   Find any vertex v of degree < 3. Let $G \leftarrow G - v$.
2. G is has a min degree 3 subgraph iff the result of 1. is nonempty.

In our construction, the interpretation is that "x removed" equates to "x is true". Suppose vertex x of the and gadget $AND(x,y,z)$ is removed. This will cause the vertex $v(x,1)$ at the top of the gadget that is joined to x to have degree 2. Thus we may remove $v(x,1)$ from $I''(C)$ since it can't be part of a min degree 3 subgraph. In turn this decreases the degree of $v(x,2)$ and $v(x,3)$ to 2. Arguing in this way we see that all of the vertices $v(x,j)$ will

need to be removed. Note that this causes $l(x, y)$ to have degree 3. Thus if one input of *and* gate is removed then the effect is to remove that half of the gadget.
In the case of an *or* gate we can similarly argue that the removal of either of the inputs will cause the whole gadget to be removed, wnd the verteces corresponding to the other input and the one corresopnding to z to have their degree reduced by 1.

Translating the gadget observations back to C it follows that if a gate is made true then the corresponding gadget can be removed from $I''(C)$. Since the given assignment is one that satisfies C it follows that any vertex u representing the output of a copy of C can be erased. Consequently, using the same reasoning we can erase all the vertices of the fanout gadgets thus dropping the degree of the vertices $v[i, 1]$ by 1. This then allows us to erase the vertices $v[i, j]$ for all i and j and thus erase all of $I''(C)$. Thus $I''(C) - \{v[i_1, 4], ..., v[i_k, 4]\}$ has no min degree 3 subgraph.

Conversely, suppose that $I''(C)$ has a a collection V' of k vertices such that $G = I''(C) - V'$ has no min degree 3 subgraph. It follows that except for variable components, V' must be disjoint from $k+1$ of the circuit copies. As the circuit copies are identical, we may suppose without loss of generality that V' has no internal circuit vertices. Now the removal of V' must cause the graph to be erased as above. This means that the removal of V' and the process above must cause *all* of the output nodes u to be removed, or else we can add the 9 nodes below of the fanout immediately below u and have a min degree 3 subgraph. Since all the output vertices must be removed by V', and not all can be in V' it follows that the circuit aboveat least one output vertex u must be removed. (Fanout gadgets can only be removed from above not below.) Since there are $2k + 1$ copies of the circuit, it follows that for one such u, its removal can only be generated by the removal of a set of variables, which we can take to be $\{v[i_1, 4], ..., v[i_m, 4]\}$. Now we can argue that since u is removed, if u corresponds to an *and* gate then *both* of its inputs must be removed, and if u is an *or* gate then at least one of its inputs must be removed. Chasing the removed inputs up the circuit, we can argue that eventually a set of vertices must be removed, and we see that this set of $\leq k$ vertices cause the circuit to be erased. The reason there are only $\leq k$ vertices are able to cause this removal is the following. Apply the removal process to the initial set V'. As the removal process discharges through the circuit the "first" time, only vertices corresponding to variables

can cause the removal of u. But if u is not removed on the first sweep then it will never be removed. As above they must correspond to a satisfying assignment for C. Since C is monotone, they can be extended to a weight k satisfying assignment for C. Thus C is satisfiable by a weight k assignment iff $I''(C)$ has a size k degree 3 subgraph annihilator, and hence MIN DEGREE 3 SUBGRAPH ANNIHILATOR is $W[P]$ hard.
(iii). This is very similar to the argument for (ii). Again, we reduce from WEIGHTED MONOTONE CIRCUIT SATISFIABILITY. This time we use the *and*, *or* and $fanout$ gates of diagram 3, where the arcs go *down*. In the construction, this time we represent the variables by pairs of vertices $x[i]$, $y[i]$ Again we form $I''(C)$ by replacing the gates in the circuit C by gadgets as above and then taking $2k+1$ copies of the for $i = 1, ..., n$. Again each *pair* that represents an output for a circuit fans out to *and* gates above each of the variable pairs. (Or more precisely, to collections of gadgets corresponding to large and gates with outpots the variables.) Note, for instance, that it requires both of the input pairs of the *and* gadget to be in the closure before the output pair will be included. Thus for the variable pairs to be included, we need all of the output pirs corresponding to the circuits to be included. Thus as in the previous argument, the original collection of $\leq 2k$ variable pairs included in V' cannot cause any more variable pairs to be included unless they first cause the inclusion of all of the circuit output pairs $first$. But as in the previous argument, this corresponds to a satisfying assignment of C.
(iv) This is similar to (iii), except that we modify the gadget so that instead of 2 incoming vertics we gat t. Diagram 3 illustrates this for an *and* gate for t=5. Here every variable is replaced by t variables.
(v) CHAIN REACTION CLOSURE is a special case of k-INDUCED 3CNF SATISFIABILITY.
(vi) This time we employ the reduction from [2]. Cook observed that if C is and circuit with input variables $x = (x_1, ..., x_n)$ and output variables $y = (y_1, ..., y_m)$ then there is a $LOGSPACE$ procedure that, on input C produces a set L of linear inequalities with the properties
(a) The inequalities include variables $X = (X_1, ..., X_n)$ and $Y = (Y_1, ..., Y_m)$.
(b) For all binary vectors $b = (b_1, ..., b_n)$ there is exactly one point that satisfies all the inequalities of L and additionally satisfies $X = b$.
(b) If c is the output of C on input b then the unique assignment that satisfies L and $X = b$ also satisfies $Y = c$.

L is said to simulate C. (The proof of this result is the following: represent *not* gates $a = \overline{b}$ by $a = 1 - b$ and $0 \leq a \leq 1$, and *and* gates $a = b \wedge c$ by $0 \leq a \leq 1,\ a \leq b,\ a \leq c$, and $b + c - 1 \leq a$.) We reduce from WEIGHTED CIRCUIT SATISFIABILITY. Let C be an instance of WEIGHTED CIRCUIT SATISFIABILITY with q input variables $x_1, ..., x_q$. Let $X_1, ..., X_q$ be rational variables. Consider the inequalities:

$$X_i \leq 0, \text{ and } k+1 \text{ copies of both } X_1 + ... + X_q \geq k, \text{ and } 0 \leq X_i \leq 1.$$

To this system we add $k + 1$ copies of the inequalities representing C with $X_1, ..., X_q$ also representing the input variables. We also add a variable Z to represent the output variable and add the inequality $Z \geq 1$ $k + 1$ times. Clearly this system is satisfiable by the deletion of k of the inequalities (which must be the $X_i \leq 0$-type) iff C has a weight k satisfying assigmment. □

Remark. There are a number of "planar" questions open for finite levels of the W-heirarchy. For instance, we do not know if PLANAR r-NORMALIZED WEIGHTED SATISFIABILITY (r fixed) is (even) $W[1]$-hard.

4 Some $W[SAT]$-complete problems

We shall need the following definition. We say that a circuit is *antimonotone* if all the inverters occur in the inputs and furthermore *all* the inputs are negated. So the sircuit is monotone save for all the inputs being negated. We have the following $W[SAT]$-completeness results to offer.

(4.1) Theorem. (Downey and Fellows) *The following are $W[SAT]$-complete*
(i) WEIGHTED MONOTONE SATISFIABILITY
(ii) WEIGHTED ANTIMONOTONE SATISFIABILITY

Proof. We only include a proof for completeness, since these results follow from [10], [11], and [12]. Let C be a boolean circuit. Following [10], we first normalize C by bringing all the inverters to the top of the circuit. As the circuit is boolean, this process involves no more than a constant increase in size using iterated applications of De Morgan's laws. Let C' be the resultant circuit. For (i) we now f.p. reduce C' to an instance of MONOTONE SATISFIABILITY. This reducton involves the basic reduction in [10] there used to reduce WEIGHTED CNF SATISFIABILITY to DOMINATING SET (which is just an instance of MONOTONE CNF SATISFIABILITY.)

We recall that this reduction went as follows. 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.

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.
$E_1 = \{c[j]a[r, s] : x_s \in C_j\}$
$E_2 = \{a[r, s]a[r, s'] : s \neq s'\}$
$E_3 = \{b[r, s, t]b[r, s, t'] : t \neq t'\}$
$E_4 = \{a[r, s]b[r, s', t] : s \neq s'\}$
$E_5 = \{b[r, s, t]d[r, s'] : s' \neq s + t \pmod{n}\}$
$E_6 = \{a[r, s]a'[r, u]\}$
$E_7 = \{b[r, s, t]b'[r, u]\}$
$E_8 = \{c[j]b[r, s, t] : \exists i\ \overline{x_i} \in C_j, s < i < s + t\}$
$E_9 = \{d[r, s]a[r', s] : r' = r + 1 \pmod{k}\}$

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 assignment represented by D satisfies X. This follows by the definition of the edge sets E_1 and E_8.

To reduce C' to a monotone circuit C'' we employ a "change of variables" based on the combinatorial reduction above. Suppose the inputs to the circuit C' received at the beginning of this step are $x[1], \ldots, x[n]$. Let X 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. Let G_X be the graph constructed for this expression as in the reduction above.

The change of variables is implemented for C' as follows. (1) Create a new input for each vertex of G_X 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 X is adjacent in G_X. (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 X is adjacent in G_X. (4) Add a new And gate to C' which takes as input the output from the bottom gate of C' and inputs corresponding to the product-of-sums expression, where the product is taken over all non-clause vertices of G_X, and the sum for a vertex u is the sum of the new inputs corresponding to the non-clause vertices in $N[u]$.

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 the verification of the reduction above. Since all of the *not* gates of C' are at the top, the circuit C'' will be monotone. Moreover it is clearly boolean, and does not involve more than a quadratic blowup of the size of C'. In fact, in [10], [11] using more intricate arguments, it is shown that the above can be implemented without changing the $weft$ of the circuit and this observation is the key to completeness results for $W[r]$. We do not need this nicety here, and thus employ a simpler argument.

We now turn to the proof of (ii). This time we use the argument of [12], Proposition 3.2. Again we give the argument for completeness. We recall that the relevant result was

$$W[t] = \text{antimonotone } W[t] \text{ for } t \text{ odd}, \ t \geq 1. \tag{4.2}$$

Again this result is proven by "hardwiring" a combinatorial reduction. This time the relevant combinatorial reduction came from the rpoof that $W[1,s] =$ antimonotone $W[1,s]$ for all $s \geq 2$. Here the reader should recall that $W[1,s]$ denoted the weighted satisfaction problem for circuits representing formulae in s-CNF form. The plan of the Downey-Fellows [12] argument is to identify a problem (RED/BLUE NONBLOCKER) that belongs to antimonotone $W[1,s]$, and then show that the problem is hard for $W[1,s]$. RED/BLUE NONBLOCKER is the parameterized problem below.

Input: A graph $G = (V, E)$ where V is partitioned into two color classes $V = V_{\text{red}} \cup V_{\text{blue}}$.
Parameter: A positive integer 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'.

The *closed neighborhood* of a vertex $u \in V$ is the set of vertices $N[u] = \{x : x \in V \text{ and } x = u \text{ or } xu \in E\}$.

It is easy to see that the restriction of RED/BLUE NONBLOCKER to graphs G of maximum degree s belongs to antimonotone $W[1,s]$ since the product-of-sums boolean expression

$$\prod_{u \in V_{\text{blue}}} \sum_{x_i \in N[u] \cap V_{\text{red}}} \neg x_i$$

has a weight k truth assignment if and only if G has size k nonblocking set. By the *weight* of a truth assignment to a set of boolean variables, we mean the number of variables assigned the value *true*.

Such an expression corresponds directly to a circuit meeting the defining conditions for antimonotone $W[1,s]$. We will refer to the restriction of RED/BLUE NONBLOCKER to graphs of maximum degree bounded by s as s-RED/BLUE NONBLOCKER. We next argue that s-RED/BLUE NONBLOCKER is complete for $W[1,s]$.

Let X be a boolean expression in conjunctive normal form with clauses of size bounded by s. Suppose X consists 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 $G = (V_{\text{red}}, V_{\text{blue}}, E)$ that has a nonblocking set of size $2k$ if and only if X is satisfied by a truth assignment of weight k.

The red vertex set V_{red} of G is the union of the following sets of vertices:
$V_1 = \{a[r_1, r_2] : 0 \leq r_1 \leq k-1, 0 \leq r_2 \leq n-1\}$
$V_2 = \{b[r_1, r_2, r_3] : 0 \leq r_1 \leq k-1, 0 \leq r_2 \leq n-1, 1 \leq r_3 \leq n-k+1\}$

The blue vertex set V_{blue} of G is the union of the following sets of vertices:
$V_3 = \{c[r_1, r_2, r_2'] : 0 \leq r_1 \leq k-1, 0 \leq r_2 < r_2' \leq n-1\}$
$V_4 = \{d[r_1, r_2, r_2', r_3, r_3'] : 0 \leq r_1 \leq k-1, 0 \leq r_2, r_2' \leq n-1, 0 \leq r_3, r_3' \leq n-1 \text{ and either } r_2 \neq r_2' \text{ or } r_3 \neq r_3'\}$
$V_5 = \{e[r_1, r_2, r_2', r_3] : 0 \leq r_1 \leq k-1, 0 \leq r_2, r_2' \leq n-1, r_2 \neq r_2', 1 \leq r_3 \leq n-k+1\}$
$V_6 = \{f[r_1, r_1', r_2, r_3] : 0 \leq r_1, r_1' \leq k-1, 0 \leq r_2 \leq n-1, 1 \leq r_3 \leq n-k+1, r_1' \neq r_2 + r_3 \bmod n\}$
$V_7 = \{g[j, j'] : 1 \leq j \leq m, 1 \leq j' \leq m_j\}$

In the desription of V_7, the integers m_j are bounded by a polynomial in n and k of degree a function of s which will be described below. Note that since s is a fixed constant independent of k, this is allowed by our definition of reduction for parameterized problems.

For convenience we distinguish the following sets of vertices.
$A(r_1) = \{a[r_1, r_2] : 0 \leq r_2 \leq n-1\}$

$B(r_1) = \{b[r_1, r_2, r_3] : 0 \leq r_2 \leq n-1, 1 \leq r_3 \leq n-k+1\}$
$B(r_1, r_2) = \{b[r_1, r_2, r_3] : 1 \leq r_3 \leq n-k+1\}$

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 for the vertex sets $V_1, ..., V_7$.
$E_1 = \{a[r_1, q]c[r_1, r_2, r_2'] : q = r_2 \text{ or } q = r_2'\}$
$E_2 = \{b[r_1, q_2, q_3]d[r_1, r_2, r_2', r_3, r_3'] : \text{ either } (q_2 = r_2 \text{ and } q_3 = r_3) \text{ or } (q_2 = r_2' \text{ and } q_3 = r_3')\}$
$E_3 = \{a[r-1, r_2]e[r_1, r_2, q, q']\}$
$E_4 = \{b[r_1, q, q']e[r_1, r_2, q, q']\}$
$E_5 = \{b[r_1, r_2, r_3]f[r_1, r_1', r_2, r_3]\}$
$E_6 = \{a[r_1 + 1 \bmod n, r_1']f[r_1, r_1', r_2, r_3]\}$

We say that a red vertex $a[r_1, r_2]$ *represents the possibility* that the boolean variable x_{r_2} may evaluate to *true* (corresponding to the possibility that $a[r_1, r_2]$ may belong to a $2k$-element nonblocking set V' in G). Similarly, we say that a red vertex $b[r_1, r_2, r_3]$ *represents the possibility* that the boolean variables $x_{r_2+1}, ..., x_{r_2+r_3-1}$ (with indices reduced mod n) may evaluate to *false*.

Suppose C is a clause of X having s literals. There are $O(n^{2s})$ distinct ways of choosing, for each literal $l \in C$, a single vertex representative of the possibility that $l = x_i$ may evaluate to *false*, in the case that l is a positive literal, or in the case that l is a negative literal $l = \neg x_i$, a representative of the possibility that x_i may evaluate to *true*. For each clause C_j of X, $j = 1, ..., m$, let $R(j, 1), R(j, 2), ..., R(j, m_j)$ be an enumeration of the distinct possibilities for such a set of representatives. We have the additional sets of edges for the clause components of G:
$E_7 = \{a[r_1, r_2]g[j, j'] : a[r_1, r_2] \in R(j, j')\}$
$E_8 = \{b[r_1, r_2, r_3]g[j, j'] : b[r_1, r_2, r_3] \in R(j, j')\}$

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 $d_r = i_{r+1(\bmod k)} - i_r \pmod n$ for $r = 0, ..., k-1$. It is straightforward to verify that the set of $2k$ vertices

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

is a nonblocking set in G.

Conversely, suppose N is a $2k$-element nonblocking set in G. It is straightforward to check that a truth assignment for X of weight k is described by

setting those variables *true* for which a vertex representative of this possibility belongs to N, and by setting all other variables to *false*.

Note that the edges of the sets E_1 (E_2) which connect pairs of distinct vertices of $A(r_1)$ ($B(r_1)$) to blue vertices of degree two, enforce that any $2k$-element nonblocking set must contain exactly one vertex from each of the sets $A(0), B(0), A(1), B(1), ..., A(k-1), B(k-1)$. The edges of E_3 and E_4 enforce (again by connections to blue vertices of degree two) that if a representative of the possibility that x_i evaluates to *true* is selected for a nonblocking set from $A(r_1)$, then a vertex in the i^{th} row of $B(r_1)$ must be selected as well, representing (consistently) the interval of variables set false (by increasing index mod$n$) until the "next" variable selected to be *true*. The edges of E_5 and E_6 insure consistency between the selection in $A(r_1)$ and the selection in $A(r_1 + 1 \bmod n)$. The edges of E_7 and E_8 insure that a consistent selection can be nonblocking if and only if it does not happen that there is a set of representatives for a clause witnessing that every literal in the clause evaluates to *false*. (There is a blue vertex for every such possible set of representatives.)

To complete the proof of (4.2) and hence Theorem 4.1 (ii), let C be a circuit of weft t for t odd, $t \geq 3$. By Theorem 4.1 of [11], we may assume that C is represented by a boolean expression E_0 that is in (alternating) product-of-sums-of-products... form (for t alternations). The first level of the circuit below the inputs consists of *And* gates (since t is odd).

Suppose the inputs to C are $x_1, ..., x_n$. Let X_1 be the boolean expression with single-literal clauses $X_1 = (x_1)(x_2)\cdots(x_n)$ and let G be the graph constructed from X_1 by the reduction in (3.4) above. Let $y_1, ..., y_z$ be new variables, one for each red vertex in G.

Let E_1 be the boolean expression

$$E_1 = \prod_{u \in (V_{\text{blue}} - V_7)} \quad \sum_{y_i \in N[u]} \neg y_i$$

and let C_1 be a circuit realizing E_1.

We modify C in the following ways:
(1) Each positive fan-out from an input x_i to C is replaced by an *And* gate receiving negated inputs from all of the new input variables y_j for which the corresponding red vertices of G represent the possibility that x_i evaluates to *false*.
(2) Each negated fan-out from an input x_i to C is replaced by an *And* gate

receiving negated inputs from all of the new input variables y_j for which the corresponding red vertices of G represent the possibility that x_i evaluates to *true*.
(3) The circuit C_1 is conjunctively combined with C at the bottommost (output) *And* gate.

The circuit C' obtained in this way accepts a weight $2k$ input vector if and only if C accepts a weight k input vector. The argument for correctness is essentially the same as for (4.1). The circuit C' has weft t after the *And* gates replacing the former inputs are coalesced with the *And* gates of the topmost large gate level (this is feasible, since t is odd). All of the input fan-out lines of C' are negated. Note that the argument again make no reference to the depth of the circuit and hence works equally well for $W[SAT]$ size circuits. □

The results of the previous section suggest natural questions such as the classification of the parameterized complexity of WEIGHTED PLANAR SATISFIABILITY and WEIGHTED PLANAR MONOTONE (CIRCUIT) SATISFIABILITY.

5 Fixed Parameter Analogues of $PSPACE$ and k-Move Games

As we have seen, classical time classes such as NP seem to split into many parameterized classes when natural parameterized versions of the problems are considered. So too the same situation seems to occur when we look at parameterized *space*. Of course, a natural parameterized space class suggests itself if we wish to consider fixed parameter space complexity.
Definition. We say that a parameterized language L is in $SLICEWISE\ PSPACE$ if there is a procedure Φ, a function f, and a constant α such that for all k,

$$\langle x,k\rangle \in L \text{ iff } \Phi(\langle x,k\rangle) \text{ accepts, and the space bound on } \Phi(\langle x,k\rangle) \text{ is } f(k)|x|^{\alpha}.$$

As the Cai *et. al.* [6] observed, one can similarly define a parameterized class $SLICEWISE$ **C** for any classical complexity class **C**. The reader should note that since there is a set in $DSPACE(|x|^{c+1})$ which in linear time can compute any language in $DSPACE(|x|^{c})$, it follows that:

Observation. *If $P = PSPACE$ then $SLICEWISE\ PSPACE = FPT$.*

One of the key goals of parameterized complexity is to give real insight into the complexity of concrete problems and the structure of P. In that light one interesting variation on the above definitions would be the class of problems in *PARAMETERIZED LOGSPACE*. That is, languages L such that for all k, $\langle x, k\rangle \in L$ is decidable in space $f(k)\log(|x|)$. While these definitions suggest some interesting analyses, we shall not pursue them here.

In this section, the main goal is to point out a very interestion connection between parameterized versions of space and the complexity of k move games. For these purposes, $SLICEWISE\ PSPACE$ and $PARAMETERIZED\ LOGSPACE$ seem too large. Of course, many game problems are known to be $PSPACE$-complete. Typically, such problems ask whether the first player to move has a winning strategy. A natural parameterized version of the problem is whether the first player has a strategy that wins within at most k moves.

Parameterized versions of some hard game problems are f. p. tractable. An example is the ALTERNATING HITTING SET game [25] [33] restricted to sets of any fixed *size* $t \geq 2$ which is $PSPACE$-complete. That is we consider the problem:

RESTRICTED ALTERNATING HITTING SET

Instance: A collection C of subsets of a set B with $|S| \leq k_1$ for all $S \in C$.

Parameter: $\langle k_1, k_2\rangle$.

Question: Does player I have a win in $\leq k_2$ moves in the following game? Players play alternatively and choose unchosen elements, until, for each $S \in C$ some member of S has been chosen. The player whose choice this happens to be wins.

(5.1) Theorem. RESTRICTED ALTERNATING HITTING SET *is (strongly) fixed parameter tractable.*

Proof. It is simplest to consider $k_1 = 2$, the analogue of the $PSPACE$ complete problem ALTERNATING VERTEX COVER. Take an edge (x, y). All vertex covers must include x or y. Try each, generating the tree of possibilities. Terminate a branch and put the cover at the leaf if a branch achieves a vertex cover. This gives a tree with at most $k_1^{k_2} = 2^{k_2}$ leaves (corresponding to posible candidates for vertex covers), at most $k_2 2^{k_2}$ vertices, and all size $\leq k_2$ covers must contain a subset occurring at one of the leaves.

Now we select k_2 additional vertices of G, not occurring at any of the leaves of the tree (we can assume V is large compared to k_2, else the problem

is easily done). Consider all possible strategies played on the subgraph induced by these at most $k_2 + k_2 2^{k_2}$ vertices. It is easy to see that player I has a winning strategy in $\leq k_2$ moves in G iff he has one in this set of strategies. □

On the other hand some problems appear not to be f. p. tractable. GENERALIZED GEOGRAPHY is a game played on a directed graph G with a distinguished start vertex [25], [33]. Players alternate choosing vertices, starting at the start vertex v_1, in such a way that the chosen vertices form, in sequence, a simple directed path in G. The first player who is unable to choose a vertex loses. Determining whether player 1 has a winning strategy in a GENERALIZED GEOGRAPHY game is $PSPACE$-complete. A good candidate for a game problem that is not f. p. tractable is *SHORT GEOGRAPHY*, in which it is asked whether player 1 has a strategy that wins a given game of GENERALIZED GEOGRAPHY in at most k moves.

In order to address such questions we introduce the classes $AW[P]$, $AW[SAT]$, and $AW[*]$, which plausibly contain problems that are not in FPT. We show that SHORT GEOGRAPHY is $AW[*]$-complete.

Like $W[SAT]$, $AW[SAT]$ is the closure under fp-reductions of a kernel problem of such a general nature that it appears not to be fixed parameter tractable. This problem is a parameterized version of Quantified Boolean Formulae, defined as follows.

Definition. *PARAMETERIZED QBFSAT* is the parameterized problem specified

Instance: A sequence $s_1, ..., s_r$ of pairwise disjoint sets of boolean variables, and a boolean formula X involving the variables in $s_1 \cup \cdots \cup s_r$.

Parameters: r, $k_1, ..., k_r$.

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 $t_1 \cup \cdots \cup t_r$ are made true, and all other variables are made false, formula X is true?

Definition. $AW[SAT]$ is the set of all problems that fp-reduce to PARAMETERIZED QBFSAT.

Clearly, an equivalent formulation of this problem is

Instance: A QBF formula $Q_1 x_1 ... Q_n x_n X$.

Parameter: $k = \langle k_1, ..., k_n \rangle$

Question: Is $Q_1^{k_1} x_1 Q_2^{k_2} x_2 ... Q_n^{k_n} x_n X$ true? (Here $\exists^{k_i} x$ is interpreted to mean "does there exist a *weight* i x such that..." and $\forall^{k_i} x$ is interpreted to mean

"for all *weight* k x..."."

Similarly, we can define the problem of PARAMETERIZED QUANTIFIED CIRCUIT SATISFIABILITY, (PARAMETERIZED QCSAT), specified by replacing the X by a circuit with the variables x as the input. So with this interpretation WEIGHTED CIRCUIT SATISFIABILITY is in PARAMETERIZED Σ_1. We have a natural result which is a partial analogue to the classical result that QBFSAT is $PSPACE$-complete. We need the following problem definition.

COMPACT TM COMPUTATION
Instance: A nondeterministic Turing machine M and a word x.
Parameter: k.
Question: Is there an accepting computation of M on input x that visits at most k work tape squares?

The following result improves the earlier work of Cai, Chen, Downey and Fellows [7] who proved that COMPACT TM COMPUTATION is $W[P]$-hard.

(5.2) Theorem. COMPACT TM COMPUTATION *is* $AW[P]$*-hard.*

Proof. Let $X, s_1, ..., s_n$ be an instance of PARAMETERIZED QCSAT with parameter $k = \langle k_1, ..., k_n \rangle$. We shall use as a subroutine the method developed by Cai *et. al.* [7] to prove $W[P]$-hardness. This proof went as follows. The reduction is from WEIGHTED MONOTONE CIRCUIT SATISFIABILITY. Let C be a circuit for which we wish to determine whether there is an input vector of weight k accepted by C. We may assume that each logic gate g of C has two inputs. In time polynomial in $|C|$ we can describe a Turing machine M sketched as follows.

M has an alphabet consisting of one letter for each input to C, and the operation of M consists of two phases. In the first phase, M makes k moves nondeterministically, writing down in the first k tape squares k symbols which represent k inputs to C set to 1. In the second phase (and visiting no other tape squares), M checks whether the the guess made in the first phase represents a vector accepted by the circuit C.

The key point is that we can structure the transition table of M to accomplish this, with the size of the table polynomial in $|C|$. To do this, we make two states q^l_{up} and q^l_{down} for each connection (or *line*) l of the circuit C. Let g be an *and* gate of C, let l be an output line of g and suppose the input lines to the gate g are l_1 and l_2. We include in the transition table for M transitions from q^l_{up} to $q^{l_1}_{up}$, from $q^{l_1}_{down}$ to $q^{l_2}_{up}$, and from $q^{l_2}_{down}$ to q^l_{down}. The significance of being a state q^l_{down} is that this represents a value of 1 for the

line l as computed by C on the input guessed in the first phase. The state q^l_{up} might be viewed as a state of *query* about the value of l for the circuit C on the input guessed in the first phase. Note that the three transitions described above for the *and* gate g thus enforce that q^l_{down} can be reached only if $q^{l_1}_{down}$ *and* $q^{l_2}_{down}$ can be reached. The appropriate transitions for an *or* gate will differ in the obvious way, i.e., we arrange that the state q^l_{down} can be reached if either of $q^{l_1}_{down}$ or $q^{l_2}_{down}$ can be reached.

If l is an input line to the the circuit C, then we encode in the state table for M a "check" (involving a scan of the k tape squares) to see if the corresponding input symbol was written during the first phase of computation. The second phase begins in the state $q^{l_{out}}_{up}$ where l_{out} is the output line of C, and the only accept state is $q^{l_{out}}_{down}$.

Note that the proof above gives a canonical way of going from a proposed collection t_i of true input variables to a Turing acceptance. Furthermore nor that there is no problem in considering only monotone circuits for our proof since the proof that WEIGHTED MONOTONE CIRCUIT SATISFIABILITY is $W[P]$ complete lifts to one that proves that PARAMETERIZED MONOTONE QCSAT is $AW[P]$-complete.

Thus to complete the proof we need to say how to introduce layers of quantifiers. Without loss of generality, we can suppose that Q_1 existential. What we do is break the work tape into n cells of size $k_1, ..., k_n$. Our first action is to write a guess for t_1 in the first k_1 squares. We build a recursive algorithm that accepts only if t_1 can be extended to a satisfying pattern according to the quantifier structure. For cell $i \geq 2$, if the quantifier corresponding to k_i is an existential one then on each sweep, in the k_i squares of *cell* i we will write a guess for the variables t_i from s_i we will be assigning true for this sweep. (We shall process the guesses in lexicographic ordering.) If the Q_i corresponding to cell i is a universal quantifier, then for each setting of the cells $1, ..., i-1$ we will cycle lexicographically through all the possibilities for t_i, that is, all the k_i element subsets of s_i. As above for each sweep we will get a setting for cells corresponding to k true variables, and we can see if M accepts. Fix a setting of $1, ..., t_{n-1}$. Recursively, note that if the last Q_n is universal then cell n will pass back a confirmation of this setting only if it successfully cycles through all possible t_n. Similarly if Q_n is existential cell n can pass a yes for this setting only if it finds a t_n. This process can be continued inductively using at most k counters, and hence we can make a

machine that uses at most $2k$ squares and accepts iff $Q_1x_1...Q_nx_nC$ is true. □

Even $AW[P]$ appears to be too large a class for our purposes. Instead, we concentrate on the class $AW[*]$ defined below.

Definition. PARAMETERIZED QBFSAT$_t$ is the restriction of PARAMETERIZED QBFSAT in which the formula part must consist of t alternating layers of conjunctions and disjunctions, with negations applied only to variables, and the main operator a conjunction. For example, if $t = 2$ then the formula must be in conjunctive normal form. $AW[t]$ is the set of all parameterized problems that fp-reduce to PARAMETERIZED QBFSAT$_t$, and $AW[*] = \bigcup_t AW[t]$.

Definition. A parameterized problem X is $AW[t]$-*complete* iff X is in $AW[t]$ and every problem in $AW[t]$ fp-reduces to X. X is $AW[*]$-*complete* iff X is in $AW[*]$ and every problem in $AW[*]$ fp-reduces to X.

(5.3) Lemma SHORT GEOGRAPHY *is in* $AW[2]$.

Proof. We use a generic reduction. Let D be a digraph with distinguished vertex v_0 upon which we shall play SHORT GEOGRAPHY with parameter k. Let $\{v_0, ..., v_n\}$ list the vertices of D and E the edge set. We shall have variables $\{p_{i,j} : 1 \leq i \leq k \wedge 0 \leq j \leq n\}$. We think of the game as a pebbling game with $p_{i,j}$ denoting the i-th pebble is on vertex v_j. We need clauses as follows:

(1) $p_{1,0}$. [Pebble 1 is on vertex v_0.]

(2) $\wedge_{1\leq i\neq j\leq q, 0\leq k\leq n}(p_{i,q} \rightarrow \overline{p_{j,q}})$ [Only one pebble per vertex.]

(3) $\wedge_{v_iv_j\notin E, 1\leq q\leq k}(p_{q,i} \rightarrow (\overline{p_{q+1,j}}\wedge\overline{p_{q-1,j}})\vee(\vee_{q'\leq q,\ q' \text{ odd},\ 0\leq t\leq n} p_{q',t}\wedge(\wedge_{v_tv_r\in E}(\vee_{q''\leq q'} p_{q'',r}))))$. [If v_iv_j not an edge then for any pebble placed on v_i the preceding pebble and the next pebble must not be on v_j unless player 1 has already won on some vertex v_t pebbled with some $q' \leq q$.]

(4) $\wedge_{2\leq j\leq k, 0\leq i\leq n}(p_{j,i} \rightarrow (\vee_{v_iv_q\in E} p_{j-1,q})\vee(\vee_{q'\leq j,\ q' \text{ odd},\ 0\leq t\leq n} p_{q't}\wedge(\wedge_{v_tv_r\in E}(\vee_{q''\leq q} p_{q'',r}))))$. [If v_i is pebbled by pebble $j \geq 2$ then some vertex adjacent to v_i must be pebbled by pebble $j-1$ unless player 1 has already won on some v_t as in (3).]

(5) $\wedge_{1\leq j\leq k-2,\ j \text{ odd},\ 0\leq i\leq n}(p_{j,i} \rightarrow (\vee(\vee_{q'\leq j, q' \text{ odd},\ 0\leq t\leq n} p_{q',t}\wedge(\wedge_{v_tv_r\in E}(\vee_{q''\leq q'} p_{q'',r})))\vee$
$((\wedge_{v_iv_j\in E} \vee_{1\leq q\leq j} p_{j,q}) \vee (\vee_{\{q,r:v_iv_q,v_qv_r\in E\}}(p_{j+1,q} \wedge p_{j+2,r})))$ [If player 1 pebbles v_i then either she wins at this play, has won at a preceding play, or player 2 pebbles a vertex v_q adjacent to v_i with the next pebble and player 1 pebbles a vertex adjacent to v_q.]

Let $P(p_{i,j} : 1 \leq i \leq k,\ 0 \leq j \leq n)$ denote the conjunction of (1)-(5) expressed in CNF form. Note that this expression has length polynomial in $\langle |D|, k\rangle$. It

is by definition in $W[2]$. The expression we then need is the following.

$$\forall y_2 \in \{p_{2,0}, ..., p_{2,n}\} \exists y_3 \in \{p_{3,0}, ..., p_{3,n}\} .... (k \text{ alternations }) P$$

This is interpreted as making the chosen variable from the set $\{p_{j,0}, ..., p_{j,n}\}$ true and the others false. Note that the form of the expression P ensures that the formula is true iff player 1 has a $\leq k$ move winning strategy. □

We prove that PARAMETERIZED QBFSAT$_t$ fp-reduces to SHORT GEOGRAPHY for every t. The reduction is actually from a restricted form of PARAMETERIZED QBFSAT$_t$.

Definition. *Unitary* PARAMETERIZED QBFSAT$_t$ is the restriction of PARAMETERIZED QBFSAT$_t$ in which the parameters $k_1, \ldots, k_r$ are all 1.

(5.4) Lemma. *For $t > 0$, Unitary* PARAMETERIZED QBFSAT$_t$ *is $AW[t]$-complete.*

Proof. The method is quite simple. Given a quantifier $\exists k$ members of$(S = \{s_1, ..., s_n\})(...)$, we can replace by $2k$ quantifiers

$$\exists x_1 \in S \forall y \in \emptyset \exists x_2 \in S \forall y \in \emptyset .... \exists x_k \in S \forall y \in \emptyset (\wedge_{i \neq j} (x_i \not\equiv x_j) \wedge ...).$$

(We treat universal quantifiers similarly.) Note that the overall parameter is doubled. We also only add an additional large *and* of *or*'s to the circuit. This expression in turn can be put into standard form by replacing the i-th occurrence of S by $S_i = \{s_1^i, ..., s_n^i\}$ and adding the expression $\wedge_{1 \leq i \leq k, j \in \{1,...,n\}} (s_j \equiv s_j^i)$. Again there is no increase in the weft of the circuit. □

(5.5) Theorem. SHORT GEOGRAPHY *is $AW[*]$-complete. Hence, $AW[*] = AW[2]$.*

Proof. We reduce PARAMETERIZED QBFSAT$_{2t}$ to SHORT GEOGRAPHY for an arbitrary $t > 0$. Let $I = (r, k_1, \ldots, k_r, s_1, \ldots, s_r, X)$ be an instance of $PARAMETERIZEDQBFSAT_{2t}$, and assume that r is odd, the leading quantifier is existential. The reduction uses ideas from Schaefer's polynomial time reduction from QBF to GENERALIZED GEOGRAPHY [33]. The graph on which the geography game is played has three parts: the choice component, the formula testing component and the literal testing component.

The choice component is similar to Shaefer's, and is designed so that player 1 chooses a member of s_1, then player 2 chooses a member of s_2, then player 1 chooses a member of s_3, etc. The chioce testing gadget is given in

diagram 4 for successive quantifier pairs $Qt_i\ Qt_{i+1}$. the gadget for Qt_i which asks us to pick *one* member from s_i consists of vertices v_i, w_i and x_j for each $x_j \in s_i$. The edges are $v_i x_j$, and $x_j w_i$ for each j. A total of $3r$ moves are made through this component, and we add two additional edges to ensure that it is player 2's move at the end, where the game enters the formula testing component.

In the formula testing component we use player 1's moves to simulate disjunctions, and player 2's moves to simulate conjunctions. A total of $2t$ moves are made through this component. Let y be bottom vertex of the choice component. Let C be a circuit representing X. Reversing the arrows, create a tree representing C with root y. We can assume that the circuit is $2t$ layers beginning with layers of, alternatively, disjunctions and then conjunctions via Downey-Fellows [10], [11]. This is why when the arrows are reversed, since y will be representing the output of a conjunction and it will be player 2's turn, player 2 will always be playing a conjunction and player 1 a disjunction. Play ends at a *literal vertex* v, corresponding to a literal in formula X, with the move being player 2's.

A literal vertex v corresponding to a positive literal x has an edge to the vertex v_x in the choice component t_i that corresponds to x. If v_x was chosen (variable x is true), then player 2 has no move, and player 1 wins. If v_x was not chosen (so x is false), then player 2 moves to v_x and wins.

If literal vertex v corresponds to a negative literal $\bar{x}$, then the literal testing component has edges (v, u_x) and (u_x, v_x), where u_x is a new vertex and v_x is the vertex corresponding to x in the choice component. Vertex u_x switches the initiative, and causes player 1 to win if v_x was not chosen, and player 2 to win if v_x was chosen.

In diagram 4 we have given an example of this construction for the formula

$$\exists x \in \{x_1, x_2, x_3\} \forall y \in \{y_1, y_2, y_3, y_4\} [(x_1 \vee y_1) \wedge (\overline{x_1} \vee \overline{y_2} \vee \overline{y_3})].$$

It is easy to see that the parity of the choices allow player 1 to win iff the formula is true since all plays begin at v_1. The total number of moves is at most $3r + 2t + 4$. Since t is fixed, the conditions of an fp-reduction are met. □

Remark. Note that the above says that there is a tradeoff between quantification and weft. It would be interesting to understand exactly why this occurs. Of course $weft$ and *quantifier complexity* are both measures of logical alternation, so perhaps this phenomenom is not too surprising.

The theorem above suggests that $AW[*]$ is perhaps the natural home of k move games. We offer one more illustration to support this idea.

We consider another game of Schaefer.

SHORT NODE KAYLES

Instance: A graph G.

Parameter: k

Question: Does I have a winning k move strategy in the following game? Players pebble a vertex not adjacent to any pebbled vertex. The first player with no play loses. I plays first.

Remark. We have stuck with the terminology of Schaefer, although the reader should think of the above as k-move ALTERNATING DOMINATING SET.

(5.6) Theorem. SHORT NODE KAYLES *is* $AW[*]$*-complete.*

Proof. First we show that the problem is $AW[*]$-hard. In view of Theorem (5.5), we only need show that the problem is $AW[2]$-hard. Let $\varphi = \exists x_k \in S_k \forall x_{k-1} \in S_{k-1} ... \exists x_1 \in S_1 (C_1 \wedge ... C_m)$ be an instance of unitary $AW[2]$. Here, we shall assume that k is odd, and $S_i = \{x_{i,1}, ..., x_{i,n_i}\}$ with $S_1 = \{x_{1,1}, x_{1,2}\}$ and $B_1 = x_{1,2} \vee x_{1,2} \vee \overline{x_{1,1}} \vee \overline{x-1,2}$. We need the vertex sets

$V = \cup_i S_i$,

$S_0 = \{z_{0,q} : 1 \leq q \leq m\}$, and

$Y_i = \{y_{i,j} : 0 \leq j \leq i-1\}$ for $1 \leq i \leq k$.

Now we need the edge sets below.

$D = \{xz_{0,q} : x$ occurs in clause $C_q\}$,

$F = \{xz_{0,q} : y \neq x,\ x, y \in S_i$ and $\overline{y}$ occurs in clause $C_q\}$, and

$G = \{y_{i,j} w : w \in ((\cup_{0 \leq p < i, p \neq j} S_p) \cup (\cup_{1 \leq r < i, r \neq j} Y_r))$.

Following ideas of Schaefer, we say that the game is played *legitimately* if the node played at move i is an element of S_{k-i+1}. We claim that if at move i a player does not play legitimately then the other player wins at the next move. Suppose that the first $k-i$ moves have been legitimate. If the player then plays illegitimately, note that he cannot have play any node from $\cup_{j \leq i+1} (S_{k-j} \cup Y_{k-j})$ as these are already dominated by pebbled vertices. Now if he plays a vertex in $S_j \cup Y_j$ for some $j < i$, then his opponent can win by playing $y_{i,j}$. (Every vertex in $(S_j \cup Y_j$ is adjacent to the illegal vertex all the rest are adjacent to either a previously played vertex or $y_{i,j}$.) If the illegitimate play is in Y_i, the only remaining possibility, it must be a $y_{i,j}$, and then the opponent can win by playing a member of S_j if $j > 0$, and either $z_{0,1}$ or $z_{0,2}$ if $j = 0$. This enforcement gadetry clearly now ensures that player

I has a winning strategy of k moves iff φ is true, as the reader can readily check.

To complete the proof, we need to establish membership of $AW[*]$. But this is essentially the same as Theorem (5.3), and is left to the reader. □

Obviously, there are a number of other k-move games that are natural candidates for $AW[*]$-completeness. We mention two. The first is ALTERNATING HITTING SET with no vertex degree restrictions. the other is
SHORT GENERALIZED HEX (See Even and Tarjan [18])
Instance. A graph G with two distinguished vertices v_1 and v_2.
Parameter. k
Question. Does I have a winning strategy in the following game. Player I plays with white pebbles and player II with black ones. Pebbles are be placed on nondistinguished vertices alternately by player I then player II. Player I wins if he can construct a path of white vertices from v_1 to v_2, in $\leq k$ moves.

6 Some Structural Results

In this section we shall relate our results to classical notions from complexity theory. Our investigations can be viewed in the context of *limited nondeterminism*. In the style of Kintala-Fischer [26], let $NP[f(n)]$ denote the class of decision problems soluable in P-time with an algorithm using only $f(n)$ bits of nondeterminism on inputs of length n.
Definition. We define the class $SUBEXPTIME(f(n))$ to be the set of languages accepted in $DTIME(p(n)2^{g(n)})$ for some function g in $o(f(n))$. (That is, $\lim_{n\to\infty} f(n)/g(n) = \infty$.)

The following results refer to the reducibility $\leq^s_T$, that is *strongly uniform* parameterized P-time reducibility.
(6.1) Theorem. *$W[P] = FPT$ iff for every P-time function f with $f(n) \geq \log n$, there is a recursive function h such that for every $L \in NP[f(n)]$ $h(L)$ computes machine M which witnesses that $L \in SUBEXPTIME(f(n))$. (We say that $NP[f(n)]$ is constructively a subset of $SUBEXPTIME(f(n))$.)*

(6.2) Corollary. *$W[P] = FPT$ implies that for all P-time f, $NP[f(n)] \subseteq SUBEXPTIME(f(n))$.*
Proof. ($\Longrightarrow$) Suppose $W[P] = FPT$. Then WEIGHTED CIRCUIT SATISFIABILITY is soluable by some procedure Φ in time $g(k)p(n)$, where p is

a polynomial, and g is some function. We can assume that g is strictly increasing, and that it grows rapidly, since any function can be bounded above by such a function.

Let $f(n) \geq \log n$ be polynomial time computable, and let α be a nondeterministic polynomial time algorithm that uses at most $f(n)$ bits of nondeterminism on inputs of length n. The following algorithm β simulates α, and we will show that β runs in time $q(n)2^{o(f(n))}$ for a polynomial q. For $k < m$, let $E_{m,k}$ be a canonical list of the size k subsets of $\{1, ..., m\}$. If s is member of $E_{m,k}$, let $N(s)$ be the index of s in list $E_{m,k}$. List $E_{m,k}$ can be chosen so that $N(s)$ can be computed in polynomial time in m. Let $[N(s)]_d$ be the d least significant bits of the binary representation of $N(s)$.
Algorithm $\beta(x)$:
$n \leftarrow |x|$.
$k \leftarrow f(n)/\log\log n$.
loop
$m \leftarrow \lceil k2^{f(n)/k} \rceil$.
$C \leftarrow$ a circuit with m inputs, and the following behavior. On an input vector v of weight k, C considers v to encode a size k subset s_v of $\{1, \ldots, m\}$, and computes $n_v = [N(s_v)]_{f(n)}$. C then simulates α on input x with guess n_v, and produces the same output as α does with that guess.
Run Φ on input (k, C) for at most $p(n)2^{f(n)/\log\log n}$ steps.
If Φ terminates within the allotted time, stop and give the same answer that it gave.
$k \leftarrow k - 1$
end loop

Since $\binom{m}{k} \geq (m/k)^k \geq 2^{f(n)}$ at each iteration, all $2^{f(n)}$ binary strings of length $f(n)$ can be passed as guesses to α, when all weight k inputs are tried. It is evident, therefore, that if $\beta(x)$ terminates, then it gives the same answer that $\alpha(x)$ does. We show that $\beta(x)$ terminates within time $q(n)2^{o(f(n))}$.

Variable m is largest for small values of k, so the last loop iteration costs the most. Since Φ terminates in time $g(k)p(n)$, the last iteration is the first one where $g(k) \leq 2^{f(n)/\log\log n}$. That is, in the last iteration $k = g^{-1}(2^{f(n)/\log\log n})$. (Here for an increasing function $z(n)$ we define $z^{-1}(n)$ to be the largest m with $z(n) \leq m$.) (Note that the initial value of k is larger than that for sufficiently rapidly growing functions g.) Considering this smallest value of k to be a function $k(n)$ of n, observe that $\lim_{n\to\infty} k(n) = \infty$.

The initial value of k is computable in polynomial time in n. The cost of producing circuit C is polynomial in m and n. But $m(n) = k(n)2^{f(n)/k(n)}$ is in $2^{o(f(n))}$, so any polynomial of m is also $2^{o(f(n))}$. The simulation of Φ is bounded in time by $p(n)2^{o(f(n))}$, so each iteration of the loop uses time at most $q(n)2^{o(f(n))}$ for some polynomial q. Since there are at most $f(n)/\log\log n$ iterations, the entire algorithm runs within time $q(n)2^{o(f(n))}$.

To complete this half of the proof, note that β is computable from α.

($\Longleftarrow$) Suppose that $NP[f(n)]$ is constructively a subset of $SUBEXPTIME(f(n))$ for every polynomial time function $f(n) \geq \log n$. Let $f_k(n) = k \log n$.

For each k, let WCS_k be the restriction of WEIGHTED CIRCUIT SATISFIABILITY to inputs of the form (x, k). Then WCS_k is in $NP[f_k(n)]$. Let α_k be a deterministic algorithm that solves WCS_k in time $p(n)2^{f_k(n)/g(n)} = p(n)n^{k/g(n)}$, where p is a polynomial and $\lim_{n\to\infty} g(n) = \infty$. By the constructive nature of the supposition, α_k is a recursive function of k. So to solve WEIGHTED CIRCUIT SATISFIABILITY on input (x, k), compute α_k, and run it on input (x, k). Call the resulting algorithm β.

Algorithm β solves WEIGHTED CIRCUIT SATISFIABILITY in time $h(k)p(n)n^{k/g(n)}$ for some function h. For $k < g(n)$, that is $h(k)q(n)$ for a polynomial q. For $k \geq g(n)$, the time is bounded by a function of k. So there is a function h' such that algorithm β runs in time $h'(k)q(n)$. □

The theorem above says nothing about the weft classes $W[t]$. There are a couple of results relating weft classes to classical notions. One idea is due to Cai and Chen [4]. They show that $W[t]$ and the class of languages L accepted by nondeterministic logarithmic time Turing machines making at most k alternations are closely related. We prove one structural result relating the soluability of SAT to $W[t]$. Let C be a class of decision circuit problems. We shall say that C is *nearly polynomial time* if there is an algorithm Φ solving C running in time $p(n)2^{o(v)}$, where $p(n)$ is a polynomial and v is the collection of input variables to C.

(6.3) Theorem. *For strong f.p. reducibility,*
(i) *For $t > 0$, $W[2t] \subseteq FPT$ implies that $SAT[2t]$ is nearly polynomial time.*
(ii) *$W[SAT] \subseteq FPT$ implies SAT is nearly polynomial time.*
(iii) *$W[P] \subseteq FPT$ implies $CIRCUIT\ SATISFIABILITY$ is nearly polynomial time.*

Proof. We prove (i), the others follow by the same method. Suppose that $W[2t]$ is FPT. Then there is a $f(k)p(n)$ time algorithm for solving WEIGHTED MONOTONE $2t$-NORMALIZED SATISFIABILITY on

instances $\langle x, k\rangle$ with $|x| = n$, for some polynomial $p(n)$ and arbitary recursive function $f(k)$. We assume that $f(k) > k$ and as in the previous proof, we let $f^{-1}(k)$ denote the largest y with $f(y) \leq k$. Furthermore we can assume that $f(k)$ is computable in time $f(k)$, so that we can easily see if $f(k) \geq y$.

Now let X be a member of $SAT[2t]$, that is, a $2t$-normalized boolean formula and consider it as a circuit with $2v$ input lines for the variables and their complements. We construct a pair (C, k) with C a circuit and k and integer to be chosen later. We ensure that C is a $2t$-normalized monotone circuit, and that X is satisfiable iff C has a weight k satisfying assignment. The reader should not that the reduction will *not* be a parameterized one. Let $s = 2^{\lceil v/k \rceil}$. The circuit C has 4 distince parts:
(1) An encoding circuit E with ks input lines.
(2) A circuit X' emulating the action of X.
(3) An enforcement circuit connected to the ks inputs.
(4) An additional *and* gate taking the outputs of X' and the constraint circuit. The single output of this *and* gate is the output for the circuit.

The ks input lines of the encoding circuit can be considered as a $k \times s$ matrix. The encoding circuit translates each row of the input into two binary numbers. Each binary numbers represent the position of the *single* 1 in that row. [The enforcement circuit will be disjunctive and will express the fact that *the input matrix has at most one 1 per row.* Since we will be looking for weight k inputs to C, this will ensure that there is exactly one 1 per row.] Each binary number has length $\log s$ and the collection of possible binary strings of length of length $\log s$ will be in exact one to one correspondence with the position of the 1 on the row. The binary numbers are complements of each other (meaning that whenever one is 1 the other is 0). One corresponds to negated variables and one corresponds to varaibles. This can all be easily achieved disjunctively using $2 \log s$ large *or* gates. For instance, for the positive variables, suppose the binary number is to be represented by $x_1 ... x_{\log s}$. We then have one *or* gate corresponding to x_1. This is connected to every second position on the row. Thus x_1 will be 1 iff the 1 on the row falls on an even square. (Similarly, if we want the negation of x_1 in the other binary string we use all the odd positions.) For x_2 we have a large *or* attached to positions $4n+3$ and $4n+4$ of the row for each n. In general, we use $2^j n + 2^{j-1} + 1, ..., 2^{j+1} n$ for each n for x_j. The idea here is that the $2v$ inpute to X are chopped into k equal pieces (sequentially) each of length $\log s$. We

note that each row has length s, so we will indeed be able to create a total of $2k \log s = 2k \times \log 2^{\lceil v/k \rceil} \geq 2v$ lines in this way. It is clear that we can use these lines as the inputs to the circuit X'. Circuit X' is the same as X except that each literal is replaced by one of the $2v$ output lines. Clearly the circuit can be massaged to be $2t$-normalized by coalescing the conjunction at the output of X' with the bottom *and*, and the disjunctions of the encoding circuit with those of the upper level of X'.

Now to complete the proof we for a given input with v variables we choose $k = f^{-1}(2^{v/\log v})$. Note that $f(k) < o(v)$. This means that the $f(k)$ factor in the solution size is acceptable. The size of C is P-time in X and ks. As k is an unbounded function of v we see that $s < 2^{o(v)}$. The running time of Φ on C is $f(k)p(|C|)$. This is $2^{o(v)}p(|X|.2^{o(v)})$. But this quantity is $p(|X|).2^{o(v)}$, giving the desired result. □

Since $v < |X|$, the reader should note that the above implies

$$\text{if } W[2] = FPT \text{ then } NP \subseteq DTIME(2^{o(n)}),$$

which would seem unlikely.

We believe that the structural issues examined above are just the tip of the iceberg in terms of parameterized complexity and its interactions with classical classes. A number of major questions remain to be resolved concerning the noncollapse/collapse of the W-hierarchy. For instance the following seem very difficult to resolve:

Open Question. Suppose that $W[t] = W[t+1]$. Does this imply any other collapse? Does collapse propogate upwards?

Open Question. Suppose that $W[t] \neq FPT$. Does that this mean that the analogue of Ladner's theorem hold? That is, does it mean that there are infinitely many problems of different parameterized degrees between $W[t]$ and FPT? More generally does density hold for the f.p. degrees of recursive sets? The answer is *yes* for strong f.p. reducibilities. (See Downey-Fellows [14])

Open Question. Is there an oracle separating the W-hierarchy?

Open Question. What is the correct notion of f.p. approximation scheme? For instance, as inspired from various issues coming from the Robertson-Seymour theorems we might ask for an algorithm for DOMINATING SET, which, when given an instance $\langle G, k \rangle$ with parameter k, either says that there is no dominating set of G of size k or gives a k-approximate one. (e.g.

one of size $2k$.) Does the existence of such a parameterized algorithm imply something like $W[2] = FPT$?

Open Question. Does every NP-complete problem have a $W[P]$ complete "parameterization"? This question involves defining the meaning of the last term.

Open Question. Every known natural language L that is $W[P]$-complete is more or less P-complete "by the slice". By this statement we mean the following: given a language L define $S(L) = \{\langle\langle x, k'\rangle k\rangle : \langle x, k'\rangle \in L \text{ and } k' \leq k\}$. [The definition of $S(L)$ makes sure that the slices of L are each coded in the one above the way that natural problems seem to be.] The question is: suppose that L is $W[P]$-complete. Is there some k such that for all $m \geq k$, the m-th slice of $S(L)$ is P-complete. As Cai *et. al.* [6] observed a yes answer seems to imply something along the lines of $P = LOGSPACE$.

Some structural issues have been discussed in Downey-Fellows [14], Cai *et. al.* [6], and Cholak-Downey [8].

7 Summary of Hardness Results in this Paper

$W[SAT]$-Complete.

WEIGHTED PLANAR SATISFIABILITY, WEIGHTED MONOTONE SATISFIABILITY, WEIGHTED ANTIMONOTONE SATISFIABILITY.

$W[P]$-Complete.

SHORT CIRCUIT SATISFIABILITY, SHORT SATISFIABILITY, k-INDUCED 3CNF SATISFIABILITY, MINIMUM AXIOM SET, WEIGHTED MONOTONE CIRCUIT SATISFIABILITY, CHAIN REACTION CLOSURE, t-THRESHOLD STARTING SET, WEIGHTED PLANAR CIRCUIT SATISFIABILITY, DEGREE 3 SUBGRAPH ANNIHILATOR, k-LINEAR INEQUALITIES.

$AW[*] = AW[2]$-Complete

SHORT GEOGRAPHY, SHORT NODE KAYLES.

$AW[P]$-hard

COMPACT TM COMPUTATION.

References

[1] K. Abrahamson, R. Downey, and M. Fellows, "Fixed Parameter Intractability II," in *Proceedings Tenth Annual Symposium on Theoretical Aspects of Computer Science,* Springer Verlag, (1993) 374-385.

[2] K. Abrahamson, J. Ellis, M. Fellows and M. Mata,"Completeness for families of Fixed Parameter Problems," *Proc. 30th ACM Foundations of Computer Science (FOCS)*, (1989), 210-215.

[3] J. Buss and J. Goldsmith,"Nondeterminism within P," SIAM J. Comput., to appear.

[4] L. Cai and J. Chen,"On the Amount of Nondeterminism and the Power of Verifying," to appear in *Proceedings of International Conference on Mathematical Foundations of Computer Science (MFCS'93),* 1993.

[5] L. Cai and J.Chen,"On Fixed-Parameter Tractability and Approximability of NP-hard Optimization Problems,"to appear in *Proceedings Israel Conference on Theoretical Computer Science, (ISTCS'93)*, 1993.

[6] L. Cai, J. Chen, R. Downey and M. Fellows, "Advice Classes of Parameterized Complexity," to appear.

[7] L. Cai, J. Chen, R. Downey, and M. Fellows, "The Parameterized Complexity of Short Computation and Factorization," to appear.

[8] P. Cholak and R. Downey, Undecidability and Definability for Parameterized Polynomial Time Reducibilities," to appear in *Logical Methods*, (ed. J. Crossley, R. Shore, M. Sweedler, and J. Remmel,) Birkhauser, to appear..

[9] R. Downey, P. Evans, and M. Fellows, "Parameterized Learning Complexity," in *Proceedings Sixth Annual Conference on Computational Learning Theory, (COLT'93)*, 1993.

[10] R. Downey and M. Fellows, "Fixed Parameter Tractability and Completeness," *Congr. Num.*, 87 (1992) 161-187.

[11] R. Downey and M. Fellows, "Fixed Parameter Tractability and Completeness I: Basic Results," to appear.

[12] R. Downey and M. Fellows, "Fixed Parameter Tractability and Completeness II: On Completeness for $W[1]$," to appear.

[13] R. Downey and M. Fellows, "Fixed Parameter Intractability (Extended Abstract)," *Proc. 7th Conf. on Structure in Complexity Theory* (1992), 36-49.

[14] R. Downey and M. Fellows, "Fixed Parameter Tractability and Completeness III: Some Structural Aspects of the W-Hierarchy," to appear in *Complexity Theory,* (ed. K. Ambos-Spies, S. Homer, and U. Schoning), Cambridge University Press.

[15] R. Downey and M. Fellows,"Parameterized Computational Feasibility," to appear in *Feasible Mathematics II*, (ed. P. Clote and J. Remmel) Birkhauser, Boston, 1993.

[16] R. Downey and M. Fellows, "Fixed Parameter Tractability," monograph in preparation.

[17] R. Downey, M. Fellows, B. Kapron, M. Hallett, and T. Wareham,"The Parameterized Complexity of Some Problems in Logic and Linguistics,"(Extended Abstract), Submitted.

[18] S. Even and R. Tarjan,"A Combinatorial Problem thet is Complete in Polynomial Space,"J. Assoc. Comput. Mach., **23**, (1976) 710-719.

[19] M. Fellows and M. Hallett,"Bandwidth is hard for $W[1]$,"in preparation.

[20] M. Fellows, M. Hallett, and H. Wareham,"DNA Physical Mapping: Three Ways Difficult,"to appear in *Proceedings First European Symposium on Algorithms,* 1993.

[21] M. Fellows and N. Koblitz,"Fixed Parameter Complexity and Cryptography,"to appear in *Proceedings of the Tenth Annual Conference on Algebraic Algorithms and Error-Correcting Codes (AAECC'93)* Springer Verlag, 1993.

[22] 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.

[23] 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.

[24] M. R. Garey and D. S. Johnson, "The Rectilinear Steiner Tree is NP-Complete,"SIAM J. Appl. Math., **32** (1977) 826-834.

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

[26] C. Kintala and P. Fischer, "Refining Nondeterminism in Relativised Polynomial Time Bounded Computations," *SIAM J. Comput.*, **9**, (1980) 46-53.

[27] P. Kolaitis and M. Thakur,"Approximation Properties of NP Minimization Problems," *Proceedings Sixth Annual Structure in Complexity Theory Conference.* IEEE Publ. (1991) 353-366.

[28] D. Lichtenstein,"Planar Formulae and their uses,"SIAM J. Computing, **11**, (1982) 329-343.

[29] C. Papadimitriou and M. Yannakakis,"On Limited Nondeterminism and the complexity of the V-C Dimension," *Proceedings Eighth Annual Structure in Complexity Conference,* IEEE Publ. (1993) 12-18

[30] K. Regan,"Finite Substructure Languages,"*Proceedings Fourth Annual Structure in Complexity Theory Conference,* (1989) 87-96.

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

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

[33] T. J. Schaefer, "Complexity of Some Two-person Perfect Information Games," *J. Comput. Sys. Sci.* 16 (1978), 185-225.