Threshold Dominating Sets and An Improved Characterization of $W[2]$

Rodney G. Downey
Department of Mathematics
P.O. Box 600
Victoria University
Wellington, New Zealand
downey@maths.vuw.ac.nz

Michael R. Fellows
Department of Computer Science
University of Victoria
Victoria, British Columbia
V8W 3P6, Canada
mfellows@csr.uvic.ca

September 27, 2008

Abstract

The Threshold Dominating Set problem is that of determining for a graph $G = (V, E)$ whether there is a subset $V' \subseteq V$ of size k, such that for each vertex $v \in V$ there are at least r elements of the closed neighborhood $N[v]$ that belong to V'. We consider the complexity of the problem parameterized by the pair (k, r). It is trivial to observe that this is hard for $W[2]$. It can also be easily shown to belong to a natural extension $W^*[2]$ of $W[2]$ defined in terms of circuit families of depth bounded by a function of the parameter. We prove membership in $W[2]$ and thus $W[2]$-completeness. Using this as a starting point, we prove that $W^*[2] = W[2]$.

Key Words: parameterized complexity, dominating sets, threshold computation, satisfiability problems, boolean circuits.

1 Introduction

The central issue in the study of parameterized computational complexity is how different parts of the input to a problem contribute to its difficulty. The following familiar problems are all concerned with the existence of sets of vertices of size k in a graph $G = (V, E)$ having various properties: Minimum Dominating Set, Feedback Vertex Set, Vertex

Cover, Independent Set (definitions can be found in [GJ79]). All of these problems are *NP*-complete, yet the parameter k seems to contribute in very different ways to the complexity of these problems. Vertex Cover and Feedback Vertex Set can be solved in time $f(k)|V|^c$ for suitable (exponential) functions f, while for Minimum Dominating Set and Independent Set the best known algorithms have a running time of $O(|V|^{ck})$.

Both outcomes are "compatible" with *NP*-hardness, yet in many applied situations (e.g., where small parameter values are of interest) we may wish to distinguish these two qualitatively different kinds of complexity behaviour. This can be viewed as one possible systematic way of "dealing with *NP*-completeness".

The framework of parameterized complexity theory pioneered in [DF95a, DF95b, DF95c, ADF95] allows us to explore this issue. Both structural results concerning parameterized complexity, and concrete complexity classifications of particular problems, are typically quite a bit more difficult than analogous classical theorems. For example, all four of the problems mentioned above can be shown to be *NP*-complete by relatively easy combinatorial reductions. In contrast, the proofs that Dominating Set is $W[2]$-complete [DF95a] and that Independent Set is $W[1]$-complete [DF95b] are quite intricate.

In this paper, we begin by examining the complexity of the following natural problem.

Threshold Dominating Set
Input: A graph $G = (V, E)$ and positive integers k and r.
Parameter: (k, r)
Question: Is there a set of vertices $V' \subseteq V$ such that: (1) $|V'| \leq k$, and (2) $\forall v \in V : |N[v] \cap V'| \geq r$? (Here $N[v]$ denotes the closed neighborhood of v, that is, $N[v] = \{u : u = v \text{ or } uv \in E\}$.)

Since this problem (for $r = 1$) includes the usual Dominating Set problem as a special case, we can observe that Threshold Dominating Set is hard for $W[2]$ by the results of [DF95a]. By showing membership in $W[2]$ we establish:

Theorem 1. Threshold Dominating Set is complete for $W[2]$.

The method used to prove Theorem 1 is applicable to the following variant of Satisfiability.

(n, k, n)-Weighted Satisfiability (WSat)
Input: A boolean expression E that is an n-product of k-sums of n-products.
Parameter: k
Question: Is there a satisfying truth assignment for the variables of E that has weight k? (That is, one that assigns exactly k variables to be *true* and the rest to be *false*.)

Note that in the definition of the above problem the parameter k plays two different roles,

in the structure of E, and also in the specification of the weight of the truth assignment. (By padding, one can easily generalize the definition to allow that the sums have arity bounded by $f(k)$ for a fixed arbitrary function f.) We prove:

Theorem 2. (n, k, n)-WSAT is complete for $W[2]$.

The $W[t]$ degrees as they are defined and characterized in [DF95a] are natural and useful because it is generally more-or-less straightforward to express a problem in logic, and this then gives membership information concerning the W hierarchy. The definition of $W[t]$ given in [DF95a] is that a parameterized language L is in $W[t]$ if it is reducible to the k-WEIGHTED CIRCUIT SATISFIABILITY problem for a family of circuits $\mathcal{C}$ satisfying:
(1) the weft of any circuit $C \in \mathcal{C}$ is at most t, where the weft of a circuit is the depth counting only *large* gates, and *small* gates have fan-in bounded by a constant c, and
(2) the depth (counting both large and small gates) of any circuit $C \in \mathcal{C}$ is bounded by a constant c'.

We can define a natural extension $W^*[t]$ by relaxing the above definition with the requirements:
(1′) the weft of any circuit $C \in \mathcal{C}$ is at most t, but any gate with fan-in bounded by an arbitrary function $h_{\mathcal{C}}(k)$ is considered *small*, and
(2′) the depth of any circuit $C \in \mathcal{C}$ is at most $h'_{\mathcal{C}}(k)$ for an arbitrary function $h'_{\mathcal{C}}$.

One can easily observe that the relaxed conditions (1′) and (2′) are equivalent to (1) and (2′). Thus the main issue in the relaxed definition is allowing the circuit depth to be a function of k.

In [DFT96] it is shown that $W^*[1] = W[1]$, and this result is used to show that the data complexity of monotone queries to relational databases is complete for $W[1]$. Our main result here is:

Theorem 3. $W^*[2] = W[2]$.

In §2 we review pertinent aspects of the framework of parameterized complexity. In §3 we prove Theorems 1 and 2 concerning THRESHOLD DOMINATING SET and (n, k, n)-WEIGHTED SATISFIABILITY. In §4 we prove the main result. §5 concludes with a discussion of some open problems.

2 Background on Parameterized Complexity

The formal framework of parameterized complexity is sketched as follows. We consider that input to a computational problem consists of two parts, one of which is expected to be

relatively small. Thus we consider a *parameterized language* L to be a set of pairs of strings, $L \subseteq \Sigma^* \times \Sigma^*$. For notational convenience, we may equivalently consider a parameterized language L to be a subset of $\Sigma^* \times N$, where for $(x, k) \in L$, k is the parameter.

The fundamental concept of the theory is the notion of *fixed-parameter tractability*. We say that a parameterized language L is *fixed parameter tractable* (FPT) if it can be determined in time $f(k)n^c$ whether $(x, k) \in L$, where c is a constant independent of the parameter k and $n = |x|$ is the size of x.

Following naturally from the concept of fixed-parameter tractability is the appropriate notion of parameterized problem reduction.

Definition. Let A, B be parameterized problems. We say that A is (uniformly many:1) *reducible* to B if there is an algorithm Φ which transforms (x, k) into $(x', g(k))$ in time $f(k)|x|^\alpha$, where $f, g : N \to N$ are arbitrary functions and α is a constant independent of k, so that $(x, k) \in A$ if and only if $(x', g(k)) \in B$.

It is easy to see that if A reduces to B and B is fixed parameter tractable then so too is A. Note that if $P = NP$ then a variety of natural parameterized NP-complete problems would be fixed-parameter tractable. Thus a completeness program is reasonable for establishing apparent fixed-parameter intractability.

The classes of parameterized problems that we define below are intuitively based on the complexity of the circuits required to check a solution. We first define circuits in which some gates have bounded fan-in and some have unrestricted fan-in. It is assumed that fan-out is never restricted.

Definition. A Boolean circuit is of *mixed type* if it consists of circuits having gates of the following kinds.
(1) *Small gates*: $\neg$ gates, $\wedge$ gates and $\vee$ gates with bounded fan-in. We will usually assume that the bound on fan-in is 2 for $\wedge$ gates and $\vee$ gates, and 1 for $\neg$ gates.
(2) *Large gates*: $\wedge$ gates and $\vee$ gates with unrestricted fan-in.

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

Definition. We say that a family of decision circuits F has *bounded depth* if there is a constant h such that every circuit in the family F has depth at most h. We say that F has *bounded weft* if there is constant t such that every circuit in the family F has weft at most t. The *weight* of a boolean vector x is the number of 1's in the vector.

Definition. Let F be a family of decision circuits. We allow that F may have many different circuits with a given number of inputs. To F we associate the parameterized circuit problem

$L_F = \{(C,k) : C \in F \text{ accepts an input vector of weight } k\}$.

Definition. A parameterized language L belongs to $W[t]$ if L reduces to the parameterized 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. A parameterized language L belongs to $W^*[t]$ if it belongs to $W[t]$ where the definition of a *small gate* has been revised to allow fan-in bounded by a fixed arbitrary function of k, and where the depth of a circuit is allowed to be a function of k as well.

Many well-known problems in various areas of computer science have been shown to be complete or hard for various levels of the W hierarchy of parameterized complexity classes [ADF95, BDFHW95, BDFW95, BF95, BFH94, CCDF94, CW95, DF95a, DF95b, DF95c, FK93, DEF93, DFT96, DFHKW94].

3 Threshold Domination and A Related Result

In this section we introduce a a fairly simple technique that appears to be quite useful. We first apply it to the Threshold Dominating Set problem defined in §1.

Theorem 1. Threshold Dominating Set is complete for $W[2]$.

Proof. Hardness for $W[2]$ follows from [DF95a] by taking $r = 1$, which yields the ordinary Dominating Set problem as a special case.

To show membership in $W[2]$, let $G = (V,E)$, k and r be given. We look at the defining property in the following way. We are allowed k choices $C_1, \ldots, C_k$ of a vertex of G. Let $S(k,r)$ denote the set of r-element subsets of $\{1, \ldots, k\}$. The question is whether we can make the k choices so that:

$$\forall u \in V, \quad \exists J \in S(k,r), \quad \forall j \in J, \quad \exists v \in N[u] \text{ with } C_j = v$$

The two inner quantifications range over index sets of size bounded by functions of k. This allows a reformulation into $\forall\exists$ form by a blowup exponential in k. The resulting formula is then entirely of the form $\forall\exists$, corresponding to the conjunctive normal form required for $W[2]$.

The details are as follows. We describe how to produce a boolean expression E in product-of-sums form that is satisfiable by a weight k truth assignment if and only if G has a k-vertex r-threshold dominating set.

The set of boolean variables for E is:

$$X = \{c[i,u] : 1 \leq i \leq k,\ u \in V\}$$

The intended meaning of $c[i,u]$ is: "the i^{th} choice of a vertex of V' is vertex u".

Let $E' = E'_1 \cdot E_2$ where

$$E'_1 = \prod_{u \in V} \sum_{J \in S(k,r)} \prod_{j \in J} \sum_{v \in N[u]} c[j,v]$$

and

$$E_2 = \prod_{i=1}^{k} \prod_{u \neq u'} (\neg c[i,u] + \neg c[i,u'])$$

The role of E_2 is to insure that, for each i, exactly one boolean variable $c[i,u]$ is assigned the value *true* in any weight k truth assignment that satisfies E'. The expression E is obtained from E' by replacing each inner sum-of-products by an equivalent product-of-sums. We leave it to the reader to check that the reduction works correctly. □

We next turn to a natural problem concerning the satisfiability of logical expressions that is easily shown to belong to $W^*[2]$, although membership in $W[2]$ is not obvious. We make a general definition that will be useful in §4.

Definition. (n,k,n)-Weighted Satisfiability (WSat) is the decision problem that takes as input a boolean expression E that is an n-product of k-sums of n-products; the question is whether there is a weight k truth assignment that satisfies E, and the parameter is k. Generalize this in the natural way for any string of n's and k's and c's where c denotes a constant. Thus (n,k,n,c)-WSat takes as input an n-product of k-sums of n-products of constant sized sums, for a fixed constant c. In order to conveniently refer to the expressions themselves, define an $\pi(n,k,n)$-*expression* to be an n-product of k-sums of n-products, and similarly define a $\sigma(n,k,n)$-*expression* to be an n-sum of k-products of n-sums, etc., where products and sums alternate, π and σ indicate what happens first, and the vector indicates the index set sizes.

Theorem 2. (n,k,n)-WSat is complete for $W[2]$.

Proof. By the antimonotone change of variables technique used in [DF95b], we can assume that all literals appearing in the (n,k,n)-WSat expression E are negated. That is, we can assume that E has the following form (assume that V is the set of boolean variables of E):

$$E = \prod_{i=1}^{n} \sum_{j=1}^{k} F(i,j)$$

where each subexpression $F(i,j)$ has the form

$$F(i,j) = \prod_{v \in V(i,j)} \neg v$$

for some set of boolean variables $V(i,j) \subseteq V$.

It is easy to see that from E we could obtain an equivalent expression by replacing each subexpression $F(i,j)$ with an expression that calculates whether k boolean variables have been set *true* in the *complementary* set of variables $V'(i,j) = V - V(i,j)$. Thus a product of negated variables can be replaced by a "threshold" calculation concerning the (monotone) set of complementary variables.

The details are as follows.

We may take the set of boolean variables and E_2 to be exactly as in the proof of Theorem 1. The meaning that we here associate to $c[i,u]$ is: "the i^{th} choice of a variable (of V) to be set *true* is variable u".

Let E_1 be the expression that results from replacing each subexpression $F(i,j)$ in E with:

$$\prod_{h=1}^{k} \sum_{u \in V'(i,j)} c[h,u]$$

We also need to insure for this problem (note that for Threshold Dominating Set the analogous issue is handled implicitly) that the k choices of variables (of V) to be set *true* are all distinct. The following subexpression will accomplish this.

$$E_3 = \prod_{1 \leq i < i' \leq k} \prod_{u \in V} (\neg c[i,u] + \neg c[i',u])$$

The reader can easily verify that the expression E' defined over the set of variables X by $E' = E_1 \cdot E_2 \cdot E_3$ has a weight k truth assignment iff E has a weight k truth assignment. Note that E' is an instance of (n,k,k,n)-WSat. By replacing the "k-sized" sums-of-products with equivalent products-of-sums as in the proof of Theorem 1, we obtain an equivalent expression in conjunctive normal form, at the cost of a blowup factor that is exponential in k.

This completes our argument that (n,k,n)-WSat is in $W[2]$, and it remains to show that it is hard for $W[2]$. (This is not entirely obvious at first glance, since *all* of the large gates are conjunctions.)

However, we may make an argument that again illustrates aspects of this technique. We reduce from the $W[2]$-hard problem Dominating Set. Let $G = (V,E)$ denote the graph of

interest. As in the proof of Theorem 1, view the property of having a dominating set of size k as whether we can make k vertex choices $C_1, \ldots, C_k$ so that:

$$\forall u \in V, \quad \exists j \in \{1, \ldots, k\}, \quad \forall v \in V - N[u], \quad C_j \neq v$$

The rest of the argument is by now routine. □

4 An Improved Characterization of $W[2]$

In this section we prove the important theorem: $W^*[2] = W[2]$. An overview of the proof is sketched as follows. In §4.1 we recall a basic lemma on the normalization of $W^*[t]$ circuits first proved in [DFT96]. In §4.2 we define an abstract notion of a *change of variables* for parameterized satisfiability problems, and extract several useful results from the previous literature on this subject. In §4.3 we prove a Lemma extending Theorem 2 that will be useful in the main argument. In §4.4 the proof of the theorem is assembled.

The reason the theorem is significant is that logic is a very good tool for establishing "upper" (class membership) bounds on parameterized complexity. One would therefore like the logical characterization of the $W[t]$ classes to provide as much expressive power as possible. While it is nontrivial to see that Threshold Dominating Set belongs to $W[2]$ with the heretofore definition, that it is complete for $W[2]$ is quite easy given the stronger characterization provided by our theorem. $W[2]$ is one of the most important and natural parameterized degrees, with quite a few well-known problems being precisely $W[2]$-complete.

4.1 Circuit Normalization

Our proof of the main theorem starts with an important lemma concerning $W^*[t]$ that says essentially that the circuits can be put in a normal form that corresponds to a certain kind of Boolean formula. The normal form is defined recursively as follows.

Definition. A Boolean expression E is in 1-alternating normal form with respect to n and k if E is either $\pi(n, k)$ or $\sigma(n, k)$. A Boolean expression E is in t-alternating normal form with respect to n and k, for $t \geq 2$, if E either is of the form:

$$E = \prod_{i=1}^{n} \sum_{j=1}^{k} E_{ij}$$

or of the form:

$$E = \sum_{i=1}^{n} \prod_{j=1}^{k} E_{ij}$$

where each sub expression E_{ij} is $(t-1)$-alternating normal with respect to n and k.

Note that the above definition can be equivalently phrased in terms of circuits in a natural way. Define a tree circuit C to be in *t-alternating normal form with respect to n and k* if it has $2t$ layers of gates, with the fan-in alternating between n and k starting with fan-in n for the output gate.

Corresponding to this form we have the following parameterized problem.

t-Alternating Weighted Satisfiability
Input: A Boolean expression E in t-alternating normal form with respect to n and k.
Parameter: k
Question: Is there a weight k truth assignment that satisfies E?

Lemma 1 (Normalization) [DFT96]. t-Alternating Weighted Satisfiability is complete for $W^*[t]$ for all $t \geq 1$.

Proof Sketch. Assume that the given circuit C has an output large gate, since an output small gate subcircuit can be removed by employing additional nondeterminism in the manner used in [DF95a]. The idea is basically to progressively analyze and "copy" the circuit C into the required form, working progressively downward from the input level. In the first step, we locate the topmost large gates. Let g denote such a gate, and let $a_1, ..., a_r$ denote the inputs to g. Suppose g is a $\wedge$ gate. Each a_i is computed by a subcircuit consisting only of small gates, and therefore the number of inputs to C on which a_i depends is bounded by a function of k. Thus, a_i can be re-expressed (at a cost that is bounded by a function of k) as a product-of-sums of total size bounded by a function of k. Thus g can be replaced by a product-of-sums "copy-gate" g', where there are at most $f(k)n$ sums (for some function f) and each sum has size bounded by some function $g(k)$. Make a copy of g' for each fan-out line of g.

For the case where g is a large $\vee$ gate, we replace each argument a_i with an equivalent sum-of-products, to obtain (for g') an $f(k)n$-sum of $g(k)$-products.

We next identify the topmost layer of large gates below the inputs to C and the copy gates created so far. Note that each copy gate has fan-out 1. We repeat the above recipe, creating two more layers of copy gates ($f'(k)n$-products of $g'(k)$-sums and $f'(k)n$-sums of $g'(k)$-products, for new functions f' and g'). Eventually the entire circuit C is replaced by the copy gates, and the resulting copy gate circuit has the required form. □

4.2 Change of Variables

The study of parameterized satisfiability problems for boolean formulae has turned out to be surprisingly challenging, and to involve a number of intricate tricks and combinatorial gadgetry. In order to avoid simply repeating these, it seems useful to articulate an abstract point of view about them, and to extract useful general results from the work that has been done to date on parameterized satisfiability [ADF95, DF95a, DF95b, DFR96, DFT96].

The following definition attempts to codify one of the most useful general tools in the study of parameterized satisfiability.

Definition. A *parameterized change of variables* for a set of *targeted subexpressions* $\mathcal{E}$ over a set of boolean variables X is a pair (r, ϕ), subject to various requirements, where:
(1) r is a *replacement function* mapping expressions in $\mathcal{E}$ to expressions over a set of variables X', and
(2) ϕ is an fpt algorithm that given X, k, and $\mathcal{E}$ computes:
(a) The set of *new variables* $X' = X'(\mathcal{E}, X, k)$.
(b) The *replacement subexpressions* $r(E_s)$ for each *subexpression* $E_s \in \mathcal{E}$
(c) A positive integer k' that is purely a function of k.
(d) An *enforcement expression* $E_0 = E_0(\mathcal{E}, X, k)$.
We require that the following conditions be met:
(1) The old variables form a subset of the new variables: $X \subseteq X'$.
(2) For every boolean expression E over X, and for every E' obtained from E by replacing some number of subexpressions E_s with $r(E_s)$ where $E_s \in \mathcal{E}$:
(a) For every weight k' truth assignment τ for X' that satisfies $E' \wedge E_0$, the restriction of τ to X has weight k and satisfies E.
(b) If there exists a weight k truth assignment to X that satisfies E, then there exists a weight k' truth assignment to X' that satisfies E'.

Intuitively, what a parameterized change of variables does for us is allow us to work with an expression E for which we wish to decide weight k satisfiability in the following way: by expanding the set of variables, and expanding k (in the parameterized sense), we can replace those parts of E that belong to $\mathcal{E}$ with replacements given by r (that are presumably simpler or more homogeneous). Yet we still have the original variables available and functioning in an "undisturbed" fashion, since $X \subseteq X'$. This means that various changes of variables can be combined (by identifying the sets X included in the new X's). It might seem that the requirements are so stringent that useful changes of variables might be hard to come by. In fact, this is not the case, and an examination of the proofs in the papers [ADF95, DF95a, DF95b, DFT96] (with some minor modifications) shows that we have the following changes of variables to work with.

Lemma 2 (Three Basic Changes of Variables). For each of the following specifications

there is a parameterized change of variables having an enforcement expression that is a product-of-sums:
(1) *Macro Change of Variables.* The set of targeted subexpressions $\mathcal{E}$ consists of monotone products of literals and antimonotone sums of literals. A monotone product can be replaced by a single positive literal, and an antimonotone sum can be replaced by a single negative literal.
(2) *Monotone Change of Variables.* The set of targeted subexpressions $\mathcal{E}$ is the set of all negative literals over the set of variables X. A negative literal can be replaced by a large product of positive literals over the set of new variables.
(3) *Antimonotone Change of Variables.* The set of targeted subexpressions $\mathcal{E}$ is the set of all positive literals over X. A positive literal can be replaced by a large product of negative literals.

Proof. The gadgetry for the three changes of variables can be found in [DF95a, DF95b]. Some slight elaboration from those proofs is necessary, but straightforward. We here allow ourselves an enforcement expression that is an arbitrary product-of-sums. Because of this extra latitude, it is easy to make the modifications necessary to meet the requirement that the set of old variables X is a subset of the new variables X' and that truth assignments to X' satisfying E' are conservative with respect to this subset in satisfying E. As the details would substantially just repeat everything in these earlier papers, we leave these to the reader. □

4.3 An Extension to Theorem 2

The problem that we consider in this section concerns boolean expressions that are products of sums of products of sums — where the last have size bounded by a constant c.

Lemma 3. (n, k, n, c)-WSAT is in $W[2]$.

Proof. Let E be the expression we start with. By means of the antimonotone change of variables we can reduce to WSAT for a boolean expression $E_1 = E_2 \wedge E_3$ where E_2 is an antimonotone (n, k, n, c, n) expression and E_3 is the enforcement (n, n) expression.

By distributing the c level upwards (at a blow-up cost of n^c, which is allowable, since c is a constant that does not depend on k), the expression E_2 can be rearranged into an equivalent antimonotone (n, k, n, c) expression.

We can now apply a macro change of variables to complete a reduction of (n, k, n, c)-WSAT to (n, k, n)-WSAT. Theorem 2 completes the proof. □

4.4 Proof of the Main Theorem

Theorem 3. $W^*[2] = W[2]$.

Proof. The proof proceeds through several steps with the eventual goal being an application of Lemma 3.

Step 1. (n, k, n, k)-Normalization.

By Lemma 1 and two macro changes of variables, we can assume that we are working with an expression E in the form $E = E_1 \cdot E_2$ where E_2 is a product-of-sums enforcement expression, and

$$E_1 = \prod_{i=1}^{n} \sum_{j=1}^{k} (E_{i,j} + F_{i,j})$$

where:
(1) Each subexpression $E_{i,j}$ is a sum of products of k literals, only one of which is positive:

$$E_{i,j} = \sum_{r=1}^{n} \left(a(i,j,r,1) \cdot \prod_{s=2}^{k} \neg a(i,j,r,s) \right)$$

with $a(i,j,r,s) \in X$ for $r = 1, ..., n$ and $s = 1, ..., k$.
(2) Each subexpression $F_{i,j}$ is a product of sums of k literals, only one of which is negative:

$$F_{i,j} = \prod_{r=1}^{n} \left(\neg b(i,j,r,1) + \sum_{s=2}^{k} b(i,j,r,s) \right)$$

with $b(i,j,r,s) \in X$ for $r = 1, ..., n$ and $s = 1, ..., k$.

Step 2. Analysis of Consequence Trees.

We begin this step working with the results of the last step, as will be true for each step of the proof. We have an expression $E = E_0 \cdot E_1$ where E_0 is a product-of-sums enforcement expression. Through the remainder of the proof, the enforcement expressions will simply accumulate and be carried along and will always be denoted E_0. The part of E on which we perform the various substitutions in this step is E_1.

To simplify notation we will always consider X to denote the set of variables for the "current" step, that is, for the expression received at the beginning of a step.

For $i \in \{1, ..., n\}$, $j \in \{1, ..., k\}$ and $v \in X$ define the *consequences of v in $E_{i,j}$* to be the family of sets:

$$\mathcal{T}_{i,j}(v) = \{\{x : \exists t \in \{1, ..., k\}\ x = a(i,j,s,t)\} : \exists s \in \{1, ..., n\}\ v = a(i,j,s,1)\}$$

and define the *consequences of* v *in* $F_{i,j}$ to be the family of sets:

$$\mathcal{U}_{i,j}(v) = \{\{x : \exists t \in \{1, ..., k\}\ x = b(i, j, s, t)\} : \exists s \in \{1, ..., n\}\ v = b(i, j, s, 1)\}$$

Let $\mathcal{T}$ denote the union of the sets $\mathcal{T}_{i,j}$ and let $\mathcal{U}$ denote the union of the sets $\mathcal{U}_{i,j}$.

Let $\mathcal{F}$ be a family of sets of size at most k over the base set X and consider the problem of finding a set X' of at most k elements of X that has nonempty intersection with each set $S \in \mathcal{F}$. By creating a k-branching tree of depth at most k, this problem can be completely analyzed, with each possible solution corresponding to a leaf of the tree. Let $\mathcal{L}_{i,j}(v)$ denote the leaf set for such an analysis of $\mathcal{T}_{i,j}(v)$ and let $\mathcal{M}_{i,j}(v)$ denote the leaf set for such an analysis of $\mathcal{U}_{i,j}(v)$. Note that these leaf sets have size bounded by a function of k.

In this step, we employ a macro change of variables in conjunction with a set of new variables that allows us a "separated" representation of the variables set to 1 in a weighted truth assignment. The set of new variables X' is:

$$X' = X'_1 \cup X'_2$$

where X'_1 are the variables introduced by the macro change of variables and

$$X'_2 = \{s[i, j] : 1 \leq i \leq k,\ 1 \leq j \leq |X|\}$$

We may assume that the set of new variables X'_1 contains the *macro* variables

$$M = \{m[S] : S \in \mathcal{T} \cup \mathcal{U}\}$$

and that the enforcement expression for the change of variables insures that for any satisfying truth assignment τ, $\tau(m[S]) = 1$ if and only if $\forall v \in S : \tau(v) = 1$ (which is the basic objective of a macro change of variables). We will also assume that a copy of the old variables X is included in X_1 (as required by the definition of a change of variables) and that this copy is denoted

$$X = \{x[i] : 1 \leq i \leq |X|\}$$

The expression E' produced by this step has the description:

$$E' = E'_0 \cdot E'_1$$

where E'_0 is the accumulating product-of-sums enforcement expression. This includes the following that has the effect of enforcing the separated representation mechanism:

$$\prod_{i=1}^{k} \sum_{j=1}^{|X|} s[i, j]$$

and

$$\prod_{1\leq i\leq k} \prod_{1\leq j<j'\leq |X|} (\neg s[i,j] + \neg s[i,j'])$$

and

$$\prod_{1\leq i<i'\leq k} \prod_{1\leq j\leq |X|} (\neg s[i,j] + \neg s[i',j])$$

and

$$\prod_{i=1}^{k} \prod_{j=1}^{|X|} (s[i,j] + \neg x[j])(\neg s[i,j] + x[j])$$

The parameter k' that accompanies E' is chosen so that there is a budget of k (weight) for the variables of X'_2. (Note the compatibility of this with the above enforcements.)

The net effect of the macro change of variables together with the separated representation mechanism is that a weight k' truth assignment can satisfy the enforcement product-of-sums expression E'_0 only if it meets these conditions:
(1) The assignment restricted to the old variables $X \subseteq X'_1$ has weight k.
(2) The variables set true in (1) are separately indicated in the k blocks of variables of X'_2.
(3) Various interesting sets of variables set true in (1) are indicated by the macro variables.
(4) The auxiliary representations (2) and (3) of (1) are forced to be consistent with (1).

The following Claims can be taken as motivation for our description of E'_1, and are important for the proof of correctness for this step. Note that the Claims are well-defined, since $X \subseteq X'$ and because of the enforcements (1-4) above.

Claim 1. A truth assignment τ of weight k' for the set of variables X' satisfies a subexpression $E_{i,j}$ of E if and only if $\exists p \in \{1, ..., k\}$ such that $\forall q \in \{1, ..., |X|\}$:

$$\neg s[p,q] + \prod_{S\in\mathcal{L}_{i,j}(x[q])} \neg m[S]$$

evaluates to 1.

Proof. Suppose that $E_{i,j}$ is satisfied by τ. This implies that one of the products of size k having one positive literal (constituting $E_{i,j}$) is satisfied. Suppose the positive literal is $x[q_0]$. Let p be the index of the block of X'_2 in which the truth of $x[q_0]$ is represented by $\tau(s[p,q_0]) = 1$. In this block of variables of X'_2, for $q \neq q_0$, $\neg s[p,q]$ evaluates to 1. To complete the proof of the Claim in this direction, we need only consider the case of $q = q_0$. Our argument fails if there is some $S \in \mathcal{L}_{i,j}(x[q_0])$ such that $\tau(m[S]) = 1$. But by the definition of the leaf sets, this implies that every k-product in the sum $E_{i,j}$ having $x[q_0]$ as the positive literal evaluates to 0, a contradiction.

Conversely, suppose p_0 satisfies the statement of the Claim. Because of the enforcements, we know that there is an index q_0 such that $s[p_0,q_0]$ evaluates to 1. Consider the k-product

terms in $E_{i,j}$ whose positive literal is $x[q_0]$. If none of these is satisfied, then by the definition of the leaf sets, for some set $S \subseteq X$ of size at most k, S has a nonempty intersection with each set in $\mathcal{L}_{i,j}(x[q_0])$ and every variable in S is assigned 1 by τ and therefore $\tau(m[S]) = 1$, a contradiction. □

Claim 2. A truth assignment τ of weight k' for the set of variables X' satisfies a subexpression $F_{i,j}$ of E if and only if $\forall p \in \{1, ..., k\}\ \exists q \in \{1, ..., |X|\}$:

$$s[p, q] \cdot \sum_{S \in \mathcal{M}_{i,j}(x[q])} m[S]$$

Proof. The proof is the DeMorgan dual of the proof of Claim 1. □

It follows from Claim 1 that each subexpression $E_{i,j}$ can be replaced by:

$$\sum_{p=1}^{k} \prod_{q=1}^{|X|} \left(\neg s[p, q] + \prod_{S \in \mathcal{L}_{i,j}(x[q])} \neg m[S] \right)$$

or by distributing:

$$\sum_{p=1}^{k} \prod_{q=1}^{|X|} \prod_{S \in \mathcal{L}_{i,j}(x[q])} (\neg s[p, q] + \neg m[S])$$

Similarly, it follows from Claim 2 that each subexpression $F_{i,j}$ can be replaced by:

$$\prod_{p=1}^{k} \sum_{q=1}^{|X|} \sum_{S \in \mathcal{M}_{i,j}(x[q])} (s[p, q] \cdot m[S])$$

Step 3. Rearrangements

We begin this step with an expression $E = E_0 \cdot E_1$ where E_0 is the accumulating product-of-sums enforcement expression and E_1 has the form:

$$E_1 = \prod_{i=1}^{n} \sum_{j=1}^{k} (E(i, j) + F(i, j))$$

where

$$E(i, j) = \sum_{p=1}^{k} \prod_{q=1}^{n} (y(i, j, p, q) + y'(i, j, p, q))$$

and

$$F(i, j) = \prod_{p=1}^{k} \sum_{q=1}^{n} (z(i, j, p, q) \cdot z'(i, j, p, q))$$

where the $y(\_,\_,\_,\_)$ and $z(\_,\_,\_,\_)$ are literals over the set of variables X.

E_1 can be rewritten as:

$$E_1 = \prod_{i=1}^{n} \left(\sum_{j=1}^{k} E(i,j) + \sum_{j=1}^{k} F(i,j) \right)$$

We have the following possibilities for further rewriting:
(1) The subexpression

$$\sum_{j=1}^{k} E(i,j)$$

can be rewritten equivalently (with permitted blow-up) as a $\sigma(k^2, n, 2)$-expression by combining the leading sums.
(2) The subexpression

$$\sum_{j=1}^{k} F(i,j)$$

can similarly be rewritten as a $\pi(k^k, kn, 2)$-expression by replacing the leading k-sum of k-products by an equivalent k^k-product of k-sums, and then combining the intermediate sums.

By rewriting E_1 (over the same set of variables) according to (1) and (2), and padding as necessary (including adjusting the parameter) we can put E_1 into the form:

$$E_1 = \prod_{i=1}^{n} \left(\sum_{j=1}^{k} G(i,j) + \prod_{j=1}^{k} H(i,j) \right)$$

where $G(i,j)$ is a $\pi(n,2)$-expression and $H(i,j)$ is a $\sigma(n,2)$-expression.

By distributing between the k-sum and the k-product this can be rewritten as:

$$E_1 = \prod_{i=1}^{n} \prod_{j=1}^{k} \left(H(i,j) + \sum_{p=1}^{k} G(i,p) \right)$$

By combining the leading products and reindexing (adjusting n and k), we have:

$$E_1 = \prod_{i=1}^{n} \left(H'(i) + \sum_{j=1}^{k} G'(i,j) \right)$$

where for each i, $H'(i)$ is a $\sigma(n,2)$-expression and for each i, j, $G'(i,j)$ is a $\pi(n,2)$-expression. (The priming simply indicates that after the reindexing and recalculation of n and k, e.g., $H'(i)$ is one of the inherited $H(i,j)$ expressions.)

Note that in some sense we are nearly done, since the above form is a $\pi(n,k,n,c)$-expression, except for the subexpressions $H'(i)$. The goal of the next step is to replace the $H'(i)$ to achieve the form of a $\pi(n,k,n,c)$-expression to which Lemma 3 can be applied.

Step 5. The Clock Change of Variables

We begin this step with $E = E_0 \cdot E_1$ where E_0 is the accumulating enforcement expression and E_1 has the form described at the end of Step 4. Our objective is to replace each subexpression $H'(i)$ with an equivalent $\sigma(k,n,2)$-expression. Note that this will achieve our goal of putting E_1 in $\pi(n,k,n,c)$ form. We employ here a new change of variables based on exactly the same gadgetry and enforcement expression as the monotone change of variables of Lemma 2, developed in [DF95a]. We simply describe here how a different set of substitutions can be made.

Consider that the set of old variables X is indexed from 0 to $|X|-1$:

$$X = \{x[i] : 0 \leq i \leq |X|-1\}$$

What the new variables X' provide is (among other things) $2k$ disjoint sets ("blocks") of variables: $Y_1, ..., Y_{2k}$, where for $i = 1, ..., k$ we have:

$$Y_i = \{y[i,j] : 0 \leq j \leq |X|-1\}$$

and for $i = k+1, ..., 2k$ we have:

$$Y_i = \{z[i,j,j'] : 0 \leq j, j' \leq |X|-1\}$$

The enforcement expression for the change of variables insures that the following conditions hold in any satisfying truth assignment:
(1) The k sets of variables $Y_1, ..., Y_k$ provide a separated representation of a weight k truth assignment to X, as in Step 2.
(2) The k sets of variables $Y_{k+1}, ..., Y_{2k}$ provide a representation of the gaps between the k variables of X set to 1 in a weight k truth assignment, as explained in [DF95a]. Thus for a truth assignment τ that satisfies the enforcement expression, $\tau(z[i,j,j']) = 1$ if and only if: $\tau(y[i,j-1]) = 1$ ("the ith choice of a variable of X set to 1 is $x[j-1]$") and $\tau(y[i+1,j'+1]) = 1$ ("the $(i+1)$th choice of a variable of X set to 1 is $x[j'+1]$") and for $s = j, ..., j'$ we have $\tau(x[s]) = 0$ ("all of the variables from $x[j]$ to $x[j']$ are set to 0").

Because of the enforcement conditions, it makes sense to say, e.g., that $y[2,3]$ *implies* $x[3]$ or that $z[3,5,8]$ *implies* $\neg x[6]$.

The validity of the substitutions we perform in this step is established by the following Claim.

Claim 3. Let F be a $\sigma(n,2)$-expression:

$$F = \sum_{i=1}^{n} (l(i) \cdot l'(i))$$

where for $i = 1, ..., n$ both $l(i)$ and $l'(i)$ are literals over the set of variables $X \subseteq X'$ according to our change of variables. For $1 \leq r, s \leq 2k$, $r \neq s$, define the set of *witness pairs*

$$\mathcal{W}(r,s) = \{(u,v) : u \in Y_r,\ v \in Y_s,\ \text{ and } \exists i \text{ such that } u \text{ implies } l(i) \text{ and } v \text{ implies } l'(i)\}$$

Modify this definition by insisting that $u = v$ if $r = s$. If τ is a truth assignment of weight k' (where k' is specified by the change of variables) that satisfies the enforcement expressions, then τ satisfies F if and only if τ satisfies:

$$F' = \sum_{r=1}^{2k} \sum_{s=1}^{2k} \prod_{(u,v) \notin \mathcal{W}(r,s)} (\neg u + \neg v)$$

where $u = v$ if $r = s$.

Proof. Suppose τ satisfies F, and let i be an index such that $l(i) \cdot l'(i)$ evaluates to 1. The enforcement for the change of variables insures that there are indices $r, s \in \{1, ..., 2k\}$ (with possibly $r = s$) and variables $u \in Y_r$ and $v \in Y_s$ such that u implies $l(i)$ and v implies $l'(i)$. Thus $(u,v) \in \mathcal{W}(r,s)$. If $r \neq s$, then since there is only one variable set to 1 in each block, $(\neg u' + \neg v')$ evaluates to 1 for every other pair $(u', v') \neq (u, v)$. Similarly if $r = s$.

Conversely, suppose τ satisfies F'. Let r and s be indices such that

$$\prod_{(u,v) \notin \mathcal{W}(r,s)} (\neg u + \neg v)$$

evaluates to 1. For convenience suppose $r \neq s$. Let (u', v') be the unique pair of variables, $u' \in Y_r$ and $v' \in Y_s$ with $\tau(u') = \tau(v') = 1$. We reach a contradiction unless $(u', v') \in \mathcal{W}(r,s)$. By the definition of the witness sets, this implies that F is satisfied. Similarly if $r = s$. □

After making the replacements justified by Claim 3, we are in a position to complete the proof of the Theorem by an application of Lemma 3. There is a small complication in that we are applying Lemma 3 to a product expression $E' = E'_0 \cdot E'_1$ where E'_0 is the accumulated enforcement $\pi(n,n)$-expression and E'_1 is the $\pi(n,k,n,2)$-expression that is the result of the last change of variables. An examination of the proof of Lemma 3 shows that this is not a problem; E'_1 can be reduced to a product-of-sums while carrying E'_0 along. □

5 Some Open Problems

The most obvious open problem arising from this work is whether $W^*[t] = W[t]$ for $t \geq 3$. We conjecture that equality holds for all t, but the proof techniques developed in [DFT96]

for t=1, and here for $t = 2$ do not seem to be adequate to deal with layers of fan-in gates that are "deep" in a circuit (Cf. Lemma 1 of §4). At present, for example, we are unable to show that the special case of (n, k, n, n)-WSat is in $W[3]$. A place to start might be the following fairly natural graph problem.

Bounded Degree Simultaneous Domination
Instance: A graph $H = (V, E)$ of maximum degree k, and a family of graphs $G_v = (X, E_v)$ indexed by V over the same set of "base set" of vertices X.
Parameter: k
Question: Is there a set of k vertices $X' \subseteq X$ such that the set V' of $v \in V$ for which X' is a dominating set in G_v, is a dominating set of H?

Is this problem complete for $W[3]$?

Acknowledgements. The authors thank Ken Regan for stimulating discussions and motivation to pursue this topic, and the second author thanks the Computer Science Department at the University of Canterbury in New Zealand for their hospitality during the preparation of this paper.

References

[ADF95] K.A. Abrahamson, R.G. Downey and M.R. Fellows. Fixed parameter tractability and completeness IV: on completeness for W[P] and PSPACE analogs. *Annals of Pure and Applied Logic* 73 (1995), 235–276.

[BDFHW95] H. Bodlaender, R.G. Downey, M.R. Fellows, M. Hallett and H.T. Wareham. Parameterized complexity analysis in computational biology. *Computer Applications in the Biosciences* 11 (1995), 49–57.

[BDFW95] H. Bodlaender, R.G. Downey, M.R. Fellows and H.T. Wareham. The parameterized complexity of the longest common subsequence problem. *Theoretical Computer Science A* 147 (1995), 31–54.

[BF95] H. Bodlaender and M.R. Fellows. On the complexity of k-processor scheduling. *Operations Research Letters* 18 (1995), 93–98.

[BFH94] H. Bodlaender, M.R. Fellows and M. Hallett. Beyond NP-completeness for problems of bounded width: hardness for the W hierarchy. *Proceedings of the ACM Symposium on the Theory of Computing (STOC)* (1994), 449–458.

[CCDF94] L. Cai, J. Chen, R.G. Downey and M.R. Fellows. On the parameterized complexity of short computation and factorization. To appear in *Arch. for Math. Logic.*

[CW95] M. Cesati and H.T. Wareham. Parameterized complexity analysis in robot motion planning. In: *Proc. 25th IEEE Intl. Conf. on Systems, Man and Cybernetics.*

[DEF93] R.G. Downey, P.A. Evans and M.R. Fellows. Parameterized learning complexity. *Proceedings of the Sixth ACM Workshop on Computational Learning Theory (COLT'93)*, 51–57,

[DF95a] R.G. Downey and M.R. Fellows. Fixed-parameter tractability and completeness I: basic theory. *SIAM Journal of Computing* 24 (1995), 873–921.

[DF95b] R.G. Downey and M.R. Fellows. Fixed-parameter tractability and completeness II: completeness for W[1]. *Theoretical Computer Science A* 141 (1995), 109–131.

[DF95c] R.G. Downey and M.R. Fellows. Parameterized computational feasibility. *Proceedings of the Second Cornell Workshop on Feasible Mathematics*, Feasible Mathematics II, P. Clote and J. Remmel (eds.), Birkhauser Boston (1995), 219–244.

[DFR96] R.G. Downey, M. Fellows and K. Regan. Parameterized circuit complexity and the W hierarchy. To appear in *Theoretical Computer Science A*.

[DFT96] R.G. Downey, M.R. Fellows and U. Taylor. The complexity of relational database queries and an improved characterization of $W[1]$. To appear in: *Proc. DMTCS 96*, Springer-Verlag, Lecture Notes in Computer Science, 1996.

[FK93] M.R. Fellows and N. Koblitz. Fixed-parameter complexity and cryptography. *Proceedings of the Tenth International Symposium on Applied Algebra, Algebraic Algorithms and Error-Correcting Codes (AAECC'93)*, Springer-Verlag, Berlin, Lecture Notes in Computer Science vol. 673 (1993), 121–131.

[GJ79] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness.* W.H. Freeman, San Francisco, 1979.

[DFHKW94] R.G. Downey, M.R. Fellows, M.T. Hallett, B.M. Kapron, and H.T. Wareham. The parameterized complexity of some problems in logic and linguistics. *Proceedings Symposium on Logical Foundations of Computer Science (LFCS)*, Springer-Verlag, Lecture Notes in Computer Science vol. 813 (1994), 89–100.