DIMACS Series in Discrete Mathematics
and Theoretical Computer Science

Descriptive Complexity and the W Hierarchy

Rodney G. Downey, Michael R. Fellows, and Kenneth W. Regan

Abstract. The classes $W[t]$ of the Downey-Fellows *W hierarchy* are defined, for each t, by fixed-parameter reductions to the weighted-assignment satisfiability problem for *weft-t* circuits. This paper proves that for each $t \geq 1$, $W[t]$ equals the closure under fixed-parameter reductions of the class of languages L definable by formulas of the form $\phi = (\exists U)\psi$, where U is a set variable and ψ is a first-order formula in $\prod_t$ prenex form. This is a fixed-parameter analogue of Fagin's well-known characterization of NP by second-order existential formulas. An equivalent form of this result states that the fixed-parameter "slices" L_k of L are definable by a family $\{ \phi_k \}$ of first-order formulas in $\sum_t$ prenex form, subject to the restriction that the quantifier blocks in ϕ_k after the leading existential block are independent of k. Whether this restriction can be removed is connected to open problems in other recent papers on the W hierarchy.

1. Parameterized Problems and the W Hierarchy

Many important and familiar problems have the general form

Instance: An object x, a natural number k.

Question: Does x have some property Π_k that depends on k?

For example, the NP-complete Clique problem asks: given an undirected graph $G = (V, E)$ and natural number k, is there a subset $U \subseteq V$ of size k that forms a clique in G? The Vertex Cover problem asks whether G has a vertex subset U of size k that covers every edge, while the Dominating Set problem asks whether there is a U of size k such that every vertex in $V \setminus U$ is adjacent to a member of U. Here k is called the *parameter*.

Formally, a *parameterized language* L is a subset of $\Sigma^* \times \mathbf{N}$. Via a simple bijective encoding from $\Sigma^* \times \mathbf{N}$ to Σ^*, one can identify a parameterized language with an ordinary language over Σ^*, but we prefer to emphasize parameterized languages as entities in their own right.

Definition 1.1 ([**DF95**]). A parameterized language L is *fixed-parameter tractable* if there is a polynomial p, a function $f : \mathbf{N} \to \mathbf{N}$, and a Turing machine

1991 *Mathematics Subject Classification.* Primary: 03D15, 68Q15, 68Q25. Secondary: 03D10, 03D20, 03D30, 03D80, 68R10.

Research supported by the New Zealand Marsden Fund for Basic Science.

Research supported by the National Science and Engineering Research Council of Canada.

Research supported by the U.S. National Science Foundation under grant CCR-9409104.

M such that on any input (x, k), M decides whether $(x, k) \in L$ within $f(k) \cdot p(|x|)$ steps. FPT stands for the class of fixed-parameter tractable languages.

This definition does not require f to be computable, and does not impose any limit on the growth rate of $f(k)$. When f is computable, L is said to belong to *strongly uniform* FPT. Languages L in FPT for which no f is computable are constructed in [**DF93**], while [**DF95**] gives *natural* problems in FPT for which no computable f is known. However, if (the ordinary language corresponding to) L belongs to P, then we can arrange that f is computable in time polynomial in the length (i.e., logarithm) of k, and further that $f(k)$ itself is polynomial in $\log k$. Thus P is contained in strongly-uniform FPT. In this paper we try to skirt the technicalities involving these and other notions of "uniformity" for FPT from [**DF95, DF93**], except to note that the issues are roughly similar to those of uniform versus non-uniform circuit classes.

If NP = P, then Clique and the other two NP-complete languages belong to FPT. The converse, however, need not hold. Indeed, Vertex Cover *does* belong to (strongly uniform) FPT, via an algorithm that runs in time $2^k \cdot O(n)$: Given a graph $G = (V, E)$, number V and E separately. To any string w in $\{0, 1\}^k$, there corresponds a size-k subset V_w of V defined as follows: To choose the ith element of V_w, let e be the least-numbered edge not yet covered. If $w_i = 0$ then add the lower-numbered vertex on e to V_w, while if $w_i = 1$, then add the higher-numbered one. Then G has a vertex cover of size k iff one of the V_w forms such a cover, and the algorithm follows. (This has been credited to several people; see [**DF94**].) Note that f is exponential in k. Quite a few other NP-complete problems, with natural parameter k, are in FPT via algorithms of time $f(k) \cdot O(n)$ through $f(k) \cdot O(n^3)$—see the compendia in [**DF95, HW96**].

The best known method for solving the parameterized Clique problem is an algorithm due to Nesetril and Poljak [**NP85**] that runs in time $O(n^{(\frac{2+\epsilon}{3})k})$, where $2 + \epsilon$ represents the exponent on the time for multiplying two $n \times n$ matrices (best known is $2.376\ldots$, see [**CW90**]). For Dominating Set we know of nothing better than the trivial $O(n^{1+k})$-time algorithm that tries all vertex subsets of size k, using $O(n)$ time per try. Many other problems listed in the appendix of [**DF95**] seem to be hard in the manner of Clique and Dominating Set. In order to compare the difficulty of problems for fixed-parameter complexity, Downey and Fellows [**DF95**] took the time-honored route of defining appropriate notions of *reducibility* and *completeness*. For the reducibility relation, we select the "many-one" rather than the "Turing" kind of FPT-reductions to talk about in this paper.

Definition 1.2 ([**DF95**]). A parameterized language A *FPT-many-one reduces* to a parameterized language B, written $A \leq_m^{fpt} B$, if there are a polynomial q, functions $f, g : \mathbf{N} \rightarrow \mathbf{N}$, and a Turing machine T such that on any input (x, k), T runs for $f(k) \cdot q(|x|)$ steps and outputs $(x', g(k))$ such that $(x, k) \in A \iff (x', g(k)) \in B$.

The reduction is *strongly uniform* if f is computable. Then (strongly uniform) FPT is closed downward under (strongly uniform) FPT reductions. Note that g is computable, and the parameter $k' = g(k)$ in the reduction does not depend on x.

Definition 1.3. For any class $\mathcal{C}$ of languages, the downward closure of $\mathcal{C}$ under strongly uniform FPT many-one reductions is denoted by $\langle \mathcal{C} \rangle_{fpt}$.

For the completeness notion, Downey and Fellows [**DF95**] defined a hierarchy of classes of parameterized languages

$$\text{FPT} \subseteq W[1] \subseteq W[2] \subseteq W[3] \subseteq \cdots \subseteq W[poly], \tag{1.1}$$

and showed that the parameterized version of Clique is complete for $W[1]$ under FPT reductions, while that of Dominating Set is complete for $W[2]$. This gives a sense in which Dominating Set is apparently harder than Clique.

This paper provides a third support for parameterized complexity theory by giving a *descriptive logical characterization* of the $W[t]$ classes, one that is analogous to the important theorems of Fagin and Stockmeyer for NP and the polynomial-time hierarchy [**Fag74, Sto76**]. There are several twists that make our results interesting and not routine. First, they relate second-order formulas to sequences of first-order formulas in a way that shows a subtle effect of the parameter k. Second, our results open new ways to understand the W classes and solve problems about them, as we show with some preliminary applications in Section 5. Third, they hint at the possibility that the W hierarchy "collapses" to $W[2]$, by analogy with a "collapse" that is implicit in Fagin's theorem itself.

Although we have not yet defined the W hierarchy, we can convey the spirit of the results by expanding the Clique and Dominating Set examples. For each k, the language L_k of graphs with a clique of size k is defined by the first-order existential formula

$$\phi_k := (\exists u_1 \dots u_k) : \bigwedge_{i,j \le k} E(u_i, u_j),$$

where $u_1 \dots u_k$ range over vertices in V. That is, we have a sequence $[\phi_k]_{k=1}^{\infty}$ of $\sum_1^{\text{FO}}$ formulas such that each ϕ_k defines the kth "slice" L_k of the parameterized language Clique. Now we can also represent Clique itself by the single second-order formula

$$\Phi = (\exists U)(\forall v, w)[(U(v) \ \wedge \ U(w) \ \wedge \ w \neq v) \rightarrow E(v, w)],$$

where U ranges over monadic relations on V—equivalently, over subsets of V. For standard languages this formula is trivial, since Φ holds for all graphs with $U = \emptyset$, but it faithfully represents the *parameterized* language Clique under the following stipulation:

> For all k, the slice L_k equals the set of strings x (encoding graphs G_x) such that Φ holds for G_x with a set U of size k.

Note that Φ has the form $(\exists U)\varphi$ where φ is a $\prod_1^{\text{FO}}$ formula, so we call Φ a $\text{SO}\exists{\cdot}\prod_1^{\text{FO}}$ formula.

Dominating Set is similarly represented by the sequence $[\psi_k]$ of $\sum_2^{\text{FO}}$ formulas with

$$\psi_k := (\exists u_1 \dots u_k) \ (\forall v) : \bigvee_{i \le k} (v = u_i \ \vee \ E(v, u_i)),$$

and by the $\text{SO}\exists \cdot \prod_2^{\text{FO}}$ formula

$$\Psi = (\exists U)(\forall v)(\exists w)[U(v) \ \vee \ (U(w) \ \wedge \ E(v, w))].$$

Thus we can write that the $W[1]$-complete language Clique belongs to $\text{SO}\exists \cdot \prod_1^{\text{FO}}$, and also to $\sum_1^{FO}$-*seq*. Similarly the $W[2]$-complete language Dominating Set belongs to $\mathrm{SO}\exists \cdot \prod_2^{\mathrm{FO}}$, and also to $\sum_2^{FO}$*-seq*. The pattern is suggestive, and indeed the first main theorem of our paper is:

THEOREM 1.4. *For all* $t \geq 1$*,* $W[t] = \langle \mathrm{SO}\exists \cdot \prod_t^{\mathrm{FO}} \rangle_{fpt}$.

One is tempted to go on to conjecture a "yes" answer to the following question:

PROBLEM 1.1. *For all* $t \geq 1$*, does* $W[t] = \langle \sum_t^{FO}\text{-}seq \rangle_{fpt}$*?*

However, this is not what we obtain. Rather, we obtain a first-order characterization of $W[t]$ by sequences of $\sum_t^{\mathrm{FO}}$ formulas ϕ_k that obey the following "niceness condition": for each of the $(t-1)$ quantifier blocks after the leading existential block, the number of variables quantified is independent of k. All $\sum_1$ families are trivially nice, so the $t = 1$ case of Problem 1.1 holds. The $\sum_2$ formulas ψ_k defining DOMINATING SET above are nice, since the "$(\forall v)$" block is the same for all of them. The second main theorem of this paper is:

THEOREM 1.5. *For all* $t \geq 1$*,* $W[t]$ *equals the* FPT*-closure of the class of parameterized languages defined by nice sequences of* $\sum_t^{\mathrm{FO}}$ *formulas.*

The "niceness" condition seems to arise by virtue of how the sequences of first-order formulas are related to the single second-order formulas in Theorem 1.4: The first existential block of a $\sum_t^{FO}$ formula ψ_k depends on k since it corresponds to the size-k set U in the $\mathrm{SO}\exists \cdot \prod_t^{\mathrm{FO}}$ formula $\Psi = (\exists U)\varphi$. The remaining $(t-1)$ quantifier blocks of ψ_k then correspond to blocks in the first-order part φ of Ψ, which are the same for all k. However, the correspondence is not so straightforward because φ has t-many quantifier blocks, so there is one left over, and even in the case of DOMINATING SET it takes some thought. In Section 5 we show how Theorem 1.5 classifies two problems related to DOMINATING SET that were previously not known to belong to $W[2]$.

In any event, the main open question of this paper becomes: What kinds of (parameterized) languages are represented by sequences of $\sum_t^{\mathrm{FO}}$ formulas with the niceness condition removed? A second important matter is raised by the link to Fagin's Theorem, which can be stated for standard languages as $\mathrm{NP} = \mathrm{SO}\exists \cdot \prod_2^{\mathrm{FO}} = \mathrm{SO}\exists \cdot \prod_t^{\mathrm{FO}}$ for all $t \geq 2$. By our Theorem 1.4, if a similar "$\mathrm{SO}\exists \cdot \prod_2^{\mathrm{FO}} = \mathrm{SO}\exists \cdot \prod_t^{\mathrm{FO}}$" collapse happens under the above stipulation for representations of parameterized languages, then $W[t] = W[2]$ for all $t \geq 2$. We note that the compendium [**HW96**] still lists no problems as $W[t]$-complete for $t \geq 3$, and indeed we have moved some problems from the "$W[2]$-hard" section into $W[2]$. For $t \leq 2$, connections similar to our Theorem 1.4 were observed by Cai and Chen [**CC93**], via the link between second-order formulas and optimization versions of NP problems that was first observed by Papadimitriou and Yannakakis [**PY91**]. We take all this to mean that the role of the parameter k deserves much further scrutiny, both at the level of NP-complete problems *and also* at the level of very-low-complexity languages defined by FO formulas.

Section 2 defines the W hierarchy, and Section 3 gives details on descriptive complexity and logic. Section 4 proves the above two theorems, and Section 5 gives applications and connections to open problems in other recent papers on the $W[t]$ classes.

2. Parameterized Circuit Complexity and the $W[t]$ Classes

Boolean circuits are said to be of *mixed type* if they may contain both *small* gates of fan-in ≤ 2 and *large* AND and/or OR gates of unbounded fan-in. We consider only *decision circuits*; i.e., those with a single output gate. The *weft* of such a circuit is the maximum number of large gates on a path from an input to the output. The n inputs are labeled by variables $x_1, \ldots, x_n$, and the *Hamming weight* $wt(x)$ of an assignment $x \in \{0,1\}^n$ equals the number of bits that are set to 1. The circuit is *monotone* if it has no NOT gates, and *anti-monotone* if all wires from an input go to a NOT gate, and these are the only NOT gates in the circuit. A pure $\sum_t$ circuit as defined by Sipser [**Sip83**] consists of t *levels* of large gates that alternate $\wedge$ and $\vee$ with a single $\vee$ gate at the top (i.e., the output), and with the bottom-level gates connected to the input gates $x_1, \ldots, x_n$ and their negations $\bar{x}_1, \ldots, \bar{x}_n$. A pure $\prod_t$ circuit is similarly defined with a large $\wedge$ gate at the output. In both cases, "pure" means that the circuit has no small gates. A Boolean expression is the same as a circuit in which each gate has fan-out 1. We call a Boolean expression t*-normalized* if it forms a pure $\prod_t$ circuit. For $t = 2$ this is the same as an expression in conjunctive normal form. For $t = 3$ this is product-of-sums-of-products (P-o-S-o-P) form; for $t = 4$ this is P-o-S-o-P-o-S form, and so on.

For all constants $h, t > 0$, the parameterized WEIGHTED CIRCUIT SATISFIABILITY problem is defined by:

$WCS(t,h)$
INSTANCE: A circuit C of weft t and overall depth $t + h$.
PARAMETER: k.
QUESTION: Does C accept some input of Hamming weight exactly k?

Then for all $t \geq 1$, $W[t]$ may be defined to be the class of parameterized languages A such that for some h, $A \leq_m^{fpt} WCS(t,h)$ (see [**BFH94**]). Also $W[poly]$ equals the class of problems that FPT many-one reduce to the problem WCS with no restriction on depth or weft. WCS is the parameterized version of the standard NP-complete CIRCUIT SATISFIABILITY problem, of which SAT is the specialization to the case where the circuit is a Boolean formula (in conjunctive normal form). An interesting aspect of the $W[\cdot]$ theory is that more-extreme special cases of the parameterized WCS problems remain complete. For all $t \geq 2$ define:

WEIGHTED t-NORMALIZED BOOLEAN EXPRESSION SATISFIABILITY
($WBES(t)$)
INSTANCE: A t-normalized Boolean expression E.
PARAMETER: k.
QUESTION: Is there a satisfying assignment to E of Hamming weight exactly k?

MONOTONE $WBES(t)$ ($MWBES(t)$)
Restriction of $WBES(t)$ to instances E that are *monotone*.

ANTI-MONOTONE $WBES(t)$ ($AWBES(t)$)
Restriction of $WBES(t)$ to instances E that are *anti-monotone*.

For $t = 1$, also define $AWBES(1,1)$ to be the restriction of $WCS(t,1)$ to instances consist of a single large AND gate, with input from a layer of binary OR gates, with the OR gates connected to negated inputs only.

THEOREM 2.1 ([**DF95, ADF93**], see also [**ADF95**]). (a) *For all* even $t \geq 2$*, MWBES*(t) *is complete for* $W[t]$ *under* $\leq_m^{fpt}$*. Hence so is WBES*(t)*.*

(b) *For all* odd $t \geq 3$*, the problem AWBES*(t) *is complete for* $W[t]$ *under* $\leq_m^{fpt}$*. Hence so is WBES*(t)*.*

(c) *The problem AWBES*$(1,1)$ *is complete for* $W[1]$ *under* $\leq_m^{fpt}$*.*

For $t = 1$, the extra level of small OR gates is necessary (unless $W[1] = \mathrm{FPT}$) [**ADF93**]. The methods there and in Section 4 in [**ADF95**] remove this layer of small gates from earlier completeness proofs for odd $t \geq 3$.

We point out one important aspect of FPT reductions that strongly governs the *size* of the objects one can produce. Suppose $A \leq_m^{fpt} WCS(t,h)$, and take the polynomial q and functions $f, g : \mathbf{N} \rightarrow \mathbf{N}$ from Definition 1.2. Since T on input (x,k) must run in time $f(k)q(n)$ $(n = |x|)$, the circuits $C_{x,k}$ it produces have size polynomial in n for fixed k, and most important, the exponent of the polynomial is independent of k. Moreover, setting $k' = g(k)$, k' becomes the Hamming weight parameter for $C_{x,k}$ and is independent of x.

DEFINITION 2.2. A *parameterized family* of circuits is a bi-indexed family of circuits $\mathcal{F} = \{ C_{n,k} \}$ such that for some functions $f, g : N \rightarrow N$ and a polynomial q, each $C_{n,k}$ has n inputs and has size at most $f(k)q(n)$. We say that such a family is *FPT-uniform* if there is an algorithm to produce the circuit $C_{n,k}$ in time $O(q(n))$.

The idea of bounded Hamming weight in the weighted circuit satisfiability problems has been very successful in classifying many problems to belong to, and be complete for, the $W[t]$ classes [**DF95, ADF93, DF94**]. Our results support the assertion that Hamming weight is a "universal parameter."

3. $W[t]$ and Descriptive Complexity

The system of first-order logic for strings used by Immerman et al. [**Imm83, Imm87, BIS90**] to characterize the *log-time hierarchy*, and to provide a robust definition for "uniform AC^0," consists of the following:

- Constants 0,1, and n; n stands for the length of the string.
- A supply of first-order variables $u, v, w, \ldots$, which range over elements of the universe $V = \{ 1, \ldots, n \}$.
- In addition to the usual quantifiers and Boolean connectives, a special unary predicate symbol $X(\cdot)$, binary predicate symbols $=, \leq$, and ternary predicate symbols $PLUS(u,v,w)$ and $TIMES(u,v,w)$.

Given a sentence ϕ with these constituents, whether a binary string x belongs to the language L_ϕ is determined as follows. Instantiate the constant symbol n to be the length of x. Let us number the bits of x beginning with 1. Then the variables $u, v, w, \ldots$ run over the domain $\{ 1, \ldots, n \}$. An atom $X(u)$ is made true by an assignment that sets u to the value i iff the ith bit of x is 1. The truth values of the other atoms depend only on the assignment to the variables, not on the input x; e.g. $TIMES(u,v,w)$ is made true by any assignment of values to u, v that makes their product equal the value assigned to w. Hence each x induces a truth value on the sentence ϕ, and $x \in L_\phi$ iff that value equals *true*.

A formula in this system that has no quantifiers is called a "matrix," and is also called both a $\sum_0^{\mathrm{FO}}$ formula and a $\prod_0^{\mathrm{FO}}$ formula. For $t \geq 1$, a $\sum_t^{\mathrm{FO}}$ formula has the form $(\exists u)\psi$, where ψ is either a $\prod_{t-1}$ formula or another $\sum_t^{\mathrm{FO}}$ formula. $\prod_t^{\mathrm{FO}}$ formulas are similarly inductively defined in terms of $\sum_{t-1}^{\mathrm{FO}}$ formulas, and are equivalent to the negations of $\sum_t^{\text{FO}}$ formulas. We may combine two or more adjacent quantifiers of the same kind into one *block*, e.g. writing $(\exists u, v)$ for $(\exists u)(\exists v)$. Then the "logical complexity" of the formula equals the minimum number of blocks needed to write (something equivalent to) the formula. The only *second-order* formulas we consider will have the form $\Phi = (\exists R)\phi$, where the second-order variable R ranges over relations on the universe V, and ϕ is a $\sum_t^{\text{FO}}$ or $\prod_t^{\text{FO}}$ formula for some t. The formula Φ is *monadic* if R is unary, so that Φ essentially quantifies over subsets of V.

The presence of the *PLUS* and *TIMES* predicates has two simplifying effects. First, it enables us to translate between formulas over *strings* and formulas over *graphs*, which have a binary relation $E(\cdot,\cdot)$ in place of $X(\cdot)$, without affecting the above notion of "logical complexity." Namely, encode an n-vertex graph G by its adjacency matrix in "row-major" order, yielding a binary string of length n^2. For each pair of vertex variables u, v, introduce a string variable w and maintain the property $w = (v-1) * n + u$. Then $E(u,v) \leftrightarrow X(w)$. This functional use of $+$ and $*$ can be represented via the *PLUS* and *TIMES* predicates with the help of some quantifiers that can be chosen to be either existential or universal. For all $t \geq 1$, a $\sum_t$ formula in the graph system is converted into a $\sum_t$ formula over strings, with extra existential quantifiers for the conversion. Thus to prove that a language encoding of a graph property belongs to $\sum_t^{\text{FO}(+,*)}$ for strings, it suffices to write a $\sum_t$ formula in the system for graphs themselves. Interestingly, it will suffice in our main results to use just E and $=$ as predicates for the graphs—*PLUS* and *TIMES* are only needed to relate the results to Immerman's systems for strings. Thus we are using "pure FO" for graphs, and this lends extra force to our writing "FO" in place of "FO$(+,*)$" in what follows.

Second, it is now known that FO$(+,*)$ is equivalent to the system FO$(BIT, \leq)$ originally used by Immerman, where $BIT(u,v)$ expresses that the vth bit of u in binary notation is a '1.' This follows from known tricks about encoding the graph of the exponential function via *PLUS* and *TIMES*, summarized by Lindell [**Lin94**] (see also [**Smo91, HP93**]), and from the result by Lindell [**Lin92**] that exponentiation suffices to simulate BIT.

Some of the features given above are redundant. For example, $=$ can be simulated via $\leq$, or alternatively $\leq$ via $=$ and *PLUS*. The constants 1 and n can be dispensed with by introducing a fresh variable v and adding the assertions '$(\forall w)[v \leq w]$,' respectively '$(\forall w)[w \leq v]$.' However, the above description is most convenient for our purposes.

These remarks apply also to second-order formulas, since the graph-to-string conversion can be bundled into the first-order parts of such formulas. For a second-order formula $\Phi = (\exists R)\phi$ in the language of graphs, R refers to an m-ary relation on the vertex set V, for some V. Our proofs going *from* $W[t]$ classes *to* formulas will produce formulas in the language of graphs. However, our proofs going *from* formulas ϕ_k or Φ *to* $W[t]$ classes will apply to formulas over any class of finite structures, including graphs and strings. This element of generality is best explained at the appropriate junctures of the proofs themselves.

DEFINITION 3.1. For all $t \geq 1$:

(a) A parameterized language L belongs to $\sum_t^{\mathrm{FO}}$-*seq* if there is a sequence of $\sum_t^{\mathrm{FO}}$ formulas ϕ_k such that for each k, ϕ_k defines the language $L_k = \{ x : (x,k) \in L \}$.

(b) The language L belongs to $\sum_t^{\mathrm{FO}}$-*nice* if in addition, all quantifier blocks in all ϕ_k after the leading existential block have size independent of k.

(c) The language L belongs to $\mathrm{SO}\exists\cdot\prod_t^{\mathrm{FO}}$ if there is an existential second-order formula Φ of the form $\Phi = (\exists R)\phi$, where R is an m-ary relation symbol (for some $m \geq 1$), ϕ is a $\prod_t$ first-order formula (naturally including the symbol R), and for all $k \geq 1$,

$$L_k = \{ x : \text{for some } \bar{R} \subseteq \{1,\ldots,n\}^m \text{ with } |\bar{R}| = k, \; x \text{ satisfies } \phi(\bar{R}) \}.$$

If Φ has the more-general form $(\exists R_1)(\exists R_2)\cdots(\exists R_\ell)\phi(R_1,\ldots,R_\ell)$, then we stipulate that L_k comprises those structures x (strings or graphs or etc.) that can satisfy $\phi(\bar{R}_1,\ldots,\bar{R}_\ell)$ for some choice of relations $\bar{R}_1,\ldots,\bar{R}_\ell$ such that $\prod_{i=1}^{\ell}|\bar{R}_i| = k$. However, *PLUS* and *TIMES* supply enough arithmetic to encode tuples of relations into a single relation, so we stay with the simpler form above. Indeed, the proofs of the two directions in our main theorems convert everything down to the case of a single monadic relation (which happens likewise for standard languages when *PLUS* and/or *TIMES* are present—see e.g. Lynch [**Lyn82**]). One interpretation of our results is that both the cardinality of a set and the Hamming weight of a string serve as a "universal parameter" for the W classes.

4. Main Results

Going from the $W[t]$ classes to the FPT-closures of the logic classes is relatively easy, since we need only find an appropriate logical specification of one $W[t]$-complete problem. The monotone and anti-monotone forms of the *WBES* problems are used crucially in this direction. We combine both the second-order and the first-order statements of our main results into one theorem and proof for each direction.

THEOREM 4.1. *For all* $t \geq 1$, $W[t] \subseteq \langle \sum_t^{\mathrm{FO}}\text{-}nice \rangle_{fpt}$, *and also* $W[t] \subseteq \langle \mathrm{SO}\exists\cdot \prod_t^{\mathrm{FO}} \rangle_{fpt}$.

PROOF. First suppose t is even. We show that the $W[t]$-complete problem MONOTONE *WBES*(t) is definable by a family of $\sum_t$ formulas in FO. An instance C of this problem is a tree with edges directed away from the root r and n sink nodes. The root represents an AND gate at level 1, and is connected to a layer of OR gates at level 2, alternating down to a layer of OR gates at level t (since t is even), which in turn is connected to the positive-only inputs. The layering enables us to avoid having to refer to the label of a gate node at all, and it is not even necessary to quantify that the nodes $u_1,\ldots,u_k$ are sinks in the graph. More importantly, because the instance C is *monotone*, it is not necessary to add to the quantifier-free matrix of the formula a term of the form $\wedge_{\,i \neq j}(u_i \neq u_j)$ saying that the u_i are all distinct. The formula ϕ_k expressing that C has a satisfying assignment of Hamming weight k is:

$$\begin{aligned}
&(\exists u_1, \ldots, u_k)\\
&\quad (\forall v_2) : E(r, v_2) \rightarrow\\
&\qquad (\exists v_3) : E(v_2, v_3) \wedge\\
&\qquad\quad (\forall v_4) : E(v_3, v_4) \rightarrow\\
&\qquad\qquad \ldots\\
&\qquad\qquad\quad (\forall v_t) : E(v_{t-1}, v_t) \rightarrow\\
&\qquad\qquad\qquad \mathrm{E}(v_t, u_1) \ \vee\ E(v_t, u_2) \ \vee\ \ldots \ \vee\ E(v_t, u_k),
\end{aligned}$$

which is equivalent to

$$\begin{aligned}
&(\exists u_1, \ldots, u_k)(\forall v_2)(\exists v_3) \ldots (\forall v_t) :\\
&\quad (\wedge_{\,i \neq j} (u_i \neq u_j)) \ \wedge\\
&\quad E(r, v_2) \rightarrow E(v_2, v_3) \ \wedge\ [E(v_3, v_4) \rightarrow E(v_4, v_5) \ \wedge\ [\ldots\\
&\quad [\ldots E(v_{t-3}, v_{t_2}) \rightarrow E(v_{t-2}, v_{t-1}) \ \wedge\ [E(v_{t-1}, v_t) \rightarrow E(v_t, u_1) \ \vee\ \ldots \ \vee\ E(v_t, u_k)]\\
&\quad] \ldots].
\end{aligned}$$

This yields a family $\{ \phi_k \}$ of $\sum_t^{\mathrm{FO}}$ sentences defining Monotone-*WBES*(t), one for each parameter k. The corresponding single second-order existential formula defining Monotone-*WBES*(t), per Definition 3.1(c), is

$$\begin{aligned}
\Phi = (\exists U)(\forall v_2)(\exists v_3) \ldots (\forall v_t)(\exists u) :&\\
&E(r, v_2) \rightarrow E(v_2, v_3) \ \wedge\ [E(v_3, v_4) \rightarrow E(v_4, v_5) \ \wedge\ [\ldots\\
&[\ldots E(v_{t-3}, v_{t_2}) \rightarrow E(v_{t-2}, v_{t-1}) \ \wedge\ [E(v_{t-1}, v_t) \rightarrow (U(u) \ \wedge\ E(v_t, u))]\\
&] \ldots].
\end{aligned}$$

Among the cases of t odd, we first consider the case $t = 1$, which involves the “extra” layer of small OR gates. An instance to Anti-Monotone *WBES*$(1,1)$ has the form $C = \ \wedge_{j=1}^{m} (\bar{x}_{j_1} \ \vee\ \bar{x}_{j_2})$, where the subscripts j_1 and j_2 run over $\{ 1, \ldots, n \}$. This has a satisfying assignment of weight k iff there is a set $S \subseteq \{ 1, \ldots, n \}$ of size k such that for all j, at least one of j_1, j_2 does not belong to S. In terms of the graph, this says that every $\vee$ node has at most one link to S. Following the above scheme with a directed tree, this yields the condition that C has a satisfying assignment of Hamming weight k iff

$$(\exists u_1, \ldots, u_k)(\forall v)[E(r, v) \implies \ \wedge_{\,i \neq j} (\bar{E}(v, u_i) \ \vee\ \bar{E}(v, u_j))].$$

However, this has $\sum_2$ form, not $\sum_1$. The key is that there are at most $\binom{n}{2}$ distinct small $\vee$ gates, and we can convert C into a DAG C' that has exactly $\binom{n}{2}$ small $\vee$ gates. Not all of these $\vee$ gates may be connected to the $\wedge$ gate—and in the analogous C' for $t \geq 3$ the graph is no longer a directed tree—but the point is that the instances of Anti-Monotone *WBES*$(t,1)$ in the proof of the main theorem of [**DF95**] can be given this C' form to begin with. Now (in the $t = 1$ case) we draw attention to the $\binom{k}{2}$ of these $\vee$ gates that have both of their input lines coming from S. The idea is to accept iff none of these gates is connected to the $\wedge$ gate at the root. It does not work to define ϕ_k to be the assertion “there exist $\binom{k}{2}$-many distinct nodes not connected to r,” because there can be sets of $\binom{k}{2}$-many $\vee$ gates that link to more than k negated inputs, in a way that defeats the assertion that a weight-k satisfying assignment exists. (For the same reason, one cannot simply regard the $\vee$ gates as a virtual input layer for the anti-monotone formula.) Instead, taking $\ell = \binom{k}{2}$, we define ϕ_k to be

$(\exists u_1, \ldots, u_k, w_1, \ldots, w_\ell)$
[These $k+\ell$ nodes are all distinct and the $u_1, \ldots, u_k$ are all sinks
and each w_i has exactly two out-links and these two belong to $\{ u_1, \ldots, u_k \}$
and $\wedge_{i=1}^{\ell} \bar{E}(r, w_i)$].

This does the job, and is a $\sum_1^{\text{FO}}$ formula. The second-order formula is actually simpler:

$$(\exists U)(\forall u_1, u_2, w) : (U(u) \rightarrow \bar{E}(u,v)) \wedge$$
$$(U(u_1) \ \wedge \ U(u_2) \ \wedge \ u_1 \neq u_2 \ \wedge \ E(w,u_1) \ \wedge \ E(w,u_2)) \rightarrow \bar{E}(r,w).$$

This says that all nodes in the set U are sinks, and all nodes w that have two out-edges into the set U are not connected to r. By above remarks on promises about the structures of the graphs in the theorem that Anti-Monotone *WBES*$(1,1)$ is $W[1]$-complete, this suffices for $W[1] \subseteq \langle \text{SO}\exists \cdot \prod_1^{\text{FO}} \rangle_{fpt}$.

In the case of odd $t \geq 3$, level t consists of large $\wedge$ gates, and the quantified v_t from this level plays the role of r in the above. It turns out not to matter whether the instance circuit has the layer of small OR gates below the inputs or not; the only change to the first-order formulas above is:

$(\exists u_1, \ldots, u_k, w_1, \ldots, w_\ell)$ [...as above and..]
$(\forall v_2) : E(r, v_2) \rightarrow$
$(\exists v_3) : E(v_2, v_3) \wedge$
$\ldots$
$(\exists v_t) : E(v_{t-1}, v_t) \wedge$
$\wedge_{i=1}^{\ell} \bar{E}(v_t, w_i)$].

This converts to $\sum_t^{\text{FO}}$ form. The second-order formula is similar to before. As remarked in the last section, all of these formulas can be converted to equivalent ones of the same quantifier structure over strings rather than graphs, with the help of the *PLUS* and *TIMES* operations. □

Remarks: The above definitions of [anti-]monotone *WBES*(t) do not include the formal definition of being t-normalized (nor of being [anti-]monotone); rather, we have treated this condition on the underlying graph as a "promise." Adding this still leaves a $\sum_t^{\text{FO}}$ formula, however. The monotonicity of the instance to *WBES*(t) for even t appears to be *vital* to this construction, and likewise the anti-monotonicity in the case of t odd. The existential second-order formulas obtained are *monadic* in the language of graphs, and can be converted to monadic formulas in the language of strings with help from the arithmetical operations *PLUS*, *TIMES*. However, for standard languages, monadic second-order existential formulas in the language of strings using just the ordering $\leq$ on string positions define only the class of regular languages (Büchi [**Büc60**]).

Now we show the converse directions:

Theorem 4.2. (a) *Let L be a parameterized language defined by a uniform family $\{ \phi_k \}$ of $\sum_t^{\text{FO}}$ formulas whose quantifier blocks after the leading existential block have size independent of k. Then $L \in W[t]$.*

(b) *Alternatively, let L be represented, per Definition 3.1(c), by a second-order existential formula* $\Phi = (\exists R)\phi$*, where* ϕ *is a single* $\prod_t^{\mathrm{FO}}$ *formula. Then* $L \in W[t]$.

PROOF. We need only show that $L \leq_m^{fpt} \mathit{WCS}(t,h)$ for some fixed h. We suppose that the formulas ϕ_k and Φ are written in the language of strings, using the fixed (interpreted) relational symbols $\{\, X(\cdot), =, \leq, \mathit{PLUS}, \mathit{TIMES} \,\}$. It will, however, be clear how to modify this proof for formulas involving other fixed relations, such as $E(\cdot,\cdot)$ for graphs. We give a general construction that takes x and k and lays out a circuit $C_{x,k}$ of size $n^{O(1)} \cdot g(k)$ and the required form for $\mathit{WCS}(t,h)$ such that $C_{x,k}$ has a weight-$f(k)$ satisfying assignment iff x makes ϕ_k true; here the functions f and g come from the family $\{\, \phi_k \,\}$ of formulas.

First we consider the case of t even, $t \geq 2$. By assumption, each formula ϕ_k has the form

$$(\exists u_1, \ldots, u_{f(k)})(\forall v_{2,1}, \ldots, v_{2,i_2}) \ldots (\forall v_{t,1}, \ldots, v_{t,i_t})\ M_k,$$

where $i_2, \ldots, i_t$ are constants independent of k, and M_k is a quantifier-free Boolean formula that can depend on k. In this even-t case, we write each M_k in conjunctive normal form (CNF); i.e., as an AND of clauses, where each clause is an OR of atoms, and each atom is a possibly-negated instance of one of the relations in $\{\, X(\cdot), =, \leq, \mathit{PLUS}, \mathit{TIMES} \,\}$. Let $c(k)$ stand for the number of clauses in ϕ_k, and $a(k)$ for the maximum number of atoms in a clause. The functions $c(k)$ and $a(k)$ may be arbitrary (computable) functions; the main point is that these quantities do not depend on the length n of x.

The circuit $C = C_{x,k}$ that we build has $nf(k)$ input nodes, thought of as $f(k)$ rows of n nodes, one row for each variable in the leading existential quantifier block. It has circuitry involving one large $\wedge$ gate of size $f(k)\binom{n}{2}$ that enforces the condition that at most one input in each row is set to '1'; this entails that any weight-$f(k)$ satisfying assignment to C must specify exactly one value in the domain $\{\, 1, \ldots, n \,\}$ for each variable.

Level 1 of the circuit is a single $\wedge$ gate with n^{i_2} fan-in lines. Each of the n^{i_2} $\vee$ gates at level 2 corresponds to a different assignment to the variables $v_{2,1}, \ldots, v_{2,i_2}$. Each of these $\vee$ gates has n^{i_3} fan-in lines, and so on in a tree down to the $\wedge$ gates at level $t-1$, each of which has fan-in lines to n^{i_t} $\wedge$ gates, called *matrix nodes*. The size of this part of the circuit is $O(n^{i_2+\cdots+i_t})$. Now we concentrate on the $\wedge$ gates for the matrix nodes at level t.

Each of these $\wedge$ gates connects to $c(k)$ *CNF clause nodes*. Each CNF clause node is an $\vee$ gate with fan-in from at most $a(k)$ *atomic nodes*. Each atomic node A is an OR gate and represents an atomic formula of the form $X(v)$, $v = w$, $v \leq w$, $\mathit{PLUS}(u,v,w)$, or $\mathit{TIMES}(u,v,w)$, where u, v, and w are variables or constants. If A involves r of the variables $u_1, \ldots, u_{f(k)}$, then A has fan-in "from all r-tuples of the *input nodes* that make A true." To explain this, first if A does not involve any of the variables $u_1, \ldots, u_{f(k)}$, then it is determined completely by the given x and the assignments to $v_{21}, \ldots, v_{t,i_t}$ along the unique path to that node from the root, and so A is filled in as a logical constant. If A has just one variable u_i from the leading block, then it is an $\vee$ gate connected to precisely those nodes in input row i whose corresponding value of u_i makes the atom true. If it has u_i and u_j where $i \neq j$, then it is an $\vee$ gate with fan-in lines to $\wedge$ gates for each pair of input values for u_i, u_j that makes the atom true. If it has three different variables, u_i,

u_j, and u_k, then there is an $\wedge$ gate for each triple of values that makes the atom true. Note that $r \leq 3$ in our string case, so the fan-in to A is at most n^3.

This completes the description of C. The two layers of AND gates for level $t-1$ and the matrix nodes can be coalesced into one layer, as can the two layers of OR gates for clause and atomic nodes. This C is a $\prod_t$ circuit plus a layer of small ANDs right at the inputs. The size of C, counting the number of gates, is bounded by

$$n^{i_2+\cdots+i_t} \cdot a(k) \cdot c(k) \cdot n^3 \cdot 3 + nf(k),$$

which in turn is bounded for all n and k by an expression of the form $g(k)n^\alpha$ with α independent of k. Thus C is in the correct form for $WCS(t,h)$, and C has a weight-$f(k)$ satisfying assignment iff x makes ϕ_k true. Hence $L \in W[t]$.

For odd t, we write the matrices M_k in disjunctive normal form (DNF). Now each matrix node is an OR gate of fan-in $t(k)$, and sends output to an OR gate at level $t-1$. Each input to a matrix node is a *term node*, which is an AND gate from $a(k)$-many atomic nodes. The problem now is that we want each atomic node A to be an AND gate.

If A involves one variable u_i from the leading block, then let A be an AND gate with *negated* input lines to all of the values in row i of the $nf(k)$ input nodes that make $A(u_i)$ *false*. Since there is "extra circuitry" enforcing that exactly one node in row i is set equal to 1, this has the same effect as before. If A uses u_i and u_j, then let A have an input line for every pair of values in those two rows that makes $A(u_i, u_j)$ false, and use an OR of two negated input wires to the nodes for each such pair. The case of three (or any fixed number of) occurrences of the u_is in an atom is handled similarly. The size of C and the rest of the analysis is the same as before. Note that in the case $t = 1$, the "extra circuitry" is an AND of small ORs (again, with negated input lines), and so this does not prevent the whole circuit from belonging to $WCS(1,h)$.

(b) In the case where we are given a second-order formula $\Phi = (\exists R)\phi$, with $\phi \in \prod_t^{\mathrm{FO}}$, ϕ has the form

$$(\forall v_{1,1}, \ldots, v_{1,i_1})(\exists v_{2,1}, \ldots, v_{2,i_2}) \ldots (Q_t v_{t,1}, \ldots, v_{t,i_t})\ M,$$

where Q_t is '$\exists$' if t is odd, Q_t is '$\forall$' if t is even, and the matrix M now has additional atoms of the form $R(v_{j_1,k_1}, \ldots, v_{j_m,k_m})$, where m is the arity of R. Now the circuit C has t, not $t-1$, levels at the top (beginning with an AND gate of fan-in n^{i_1} at the output) for the first-order variables. Things are actually simpler here insofar as there is no dependence of any first-order variables on the parameter k, and all assignments are determined by the path through the first t levels. Hence each atom other than $R(\cdots)$ is filled in as a constant. Only the nodes for each occurrence of $R(\cdots)$ in M are connected to the input nodes. There are n^m input nodes, one for each possible sequence of m values $a_1, \ldots, a_n$ to the arguments of R. Every assignment (of weight k) to these nodes determines a unique relation R (of size k). So for each occurrence of $R(v_{j_1,k_1}, \ldots, v_{j_m,k_m})$ below a given level-t node, we read off the values $a_1, \ldots, a_n$ to $v_{j_1,k_1}, \ldots, v_{j_m,k_m}$ from the path to that level-t node, and send a single input line to the corresponding input node. $\square$

5. Applications and Conclusions

For an example of how these theorems can be used to classify problems in the $W[t]$ hierarchy, consider the following.

DOMINATING THRESHOLD SET
INSTANCE: An undirected graph $G = (V, E)$, and integers $k, r > 0$.
PARAMETER: k, r.
QUESTION: Is there a set U of at most k vertices such that for all $v \in V$, the neighborhood of v contains at least r elements of U?

This is defined by the family $\{ \phi_{k,r} \}$ of $\sum_2^{\mathrm{FO}}$ formulas given by

$$\phi_{k,r} = (\exists u_1, u_2, \ldots, u_k)(\forall v)\, M,$$

where

$$M = (\vee_{\text{distinct}}\ i_1, \ldots, i_r \in \{1, \ldots, k\})(\wedge_{q=1}^{r})[v = u_{i_q} \ \vee\ E(v, u_{i_q})].$$

Hence DOMINATING THRESHOLD SET belongs to $W[2]$. Since it was previously known to be $W[2]$-hard under $\leq_m^{fpt}$, this classifies it as $W[2]$-complete. The same result was obtained by Downey, Fellows, and Koblitz [**DF96, DFK96**] by other techniques.

For a second example, from the section of the compendium [**HW96**] titled "$W[2]$-Hard," consider

DOMINATING CLIQUE
INSTANCE: An undirected graph $G = (V, E)$, and an integer $k > 0$.
PARAMETER: k.
QUESTION: Is there a set U of at most k vertices that forms a clique in G, such that for all $v \in V$, there is an edge from v to a node in U?

This is defined by the sequence $[\phi_k]$ with

$$\phi_k = (\exists u_1, u_2, \ldots, u_k)(\forall v)\, M,$$

where

$$M = [\wedge_{1 \leq i < j \leq k} E(u_i, u_j)] \bigwedge [\vee_{1 \leq i \leq k} E(v, u_i)].$$

Since this is also a "nice" sequence of formulas, DOMINATING CLIQUE belongs to $W[2]$, and since it is $W[2]$-hard, this classifies it as $W[2]$-complete.

Finally, the problem of "internal quantifier blocks that depend on k" provided part of the motivation for the following work in [**DF96**]. For any sequence of numbers $(a_1, a_2, \ldots, a_r)$, where we intend each a_i to be either "k" or "n", define

$(a_1, a_2, \ldots, a_r)$-WEIGHTED SATISFIABILITY
INSTANCE: A Boolean expression F that consists of an AND of a_1-many ORs of a_2-many ANDs of … of a_r literals.
PARAMETER: k.
QUESTION: Is there an assignment a of weight exactly k that satisfies F?

In particular, (n, k, n)-WSAT is the case where F is an n-ary AND of k-ary ORs, with each of the k-many inputs to each OR being an AND of n literals. As before, if all of the literals are positive, the formula F is *monotone*; if they are all negated, F is *anti-monotone*. Downey and Fellows [**DF96**], starting with a more-general circuit definition of a classes $W^*[t]$ that extend the definition of $W[t]$, show that for all $t \geq t$, $(n,k)^t$-WSAT is complete for $W^*[t]$ under $\leq_m^{fpt}$, so $W^*[t] = \langle (n,k)^t\text{-WSAT}\rangle_{fpt}$. By techniques similar to those from [**DF95**] that we referenced in Section 3, they show that anti-monotone (n,k,n,k)-WSAT remains $W^*[2]$-complete. Finally they show that anti-monotone (n,k,n,k)-WSAT is in $W[2]$, thus proving $W^*[2] = W[2]$. That $W^*[1] = W[1]$ was shown in [**DFT96**]. They inquire whether $W^*[t] = W[t]$ for all t.

We can show that anti-monotone (n,k,n)-WSAT and (n,k,n,k)-WSAT belong to $W[2]$, by applying the proof (if not the statement) of Theorem 4.2. Let us represent an anti-monotone (n,k,n)-WSAT formula F by a leveled graph as before, with n nodes for the negated inputs at level 0, and a single node at level 3 for the n-ary output AND gate. Intuitively let "w" range over the k-ary OR gates at level 2, "v" over the n-ary AND gates at level 1, and "u" over the inputs. Then the formula F has a satisfying assignment of Hamming weight k if and only if

$$(\exists u_1,\ldots,u_k)(\forall w)(\text{"let } v_1,\ldots,v_k \text{ be the } k \text{ nodes connected to } w\text{"})M,$$

where M is the Boolean matrix $\vee_{i=1}^{k} \wedge_{j=1}^{k}[\neg E(v_i,u_j)]$. Except for the funny "let" construct, this yields a "nice" sequence of $\sum_2^{\text{FO}}$ formulas, since the $(\forall w)$ part is unchanged. Because the $v_1,\ldots,v_k$ are uniquely determined once w is fixed, the presence of the "let" does not affect the counting in the proof of Theorem 4.2, and the conclusion that (n,k,n)-WSAT belongs to $W[2]$ follows. The case of (n,k,n,k)-WSAT is analogous to how the "extra" level of OR gates was handled in the $t=1$ case of Theorem 4.1—both the leading existential block and the matrix become larger and more complicated, but the form is the same.

For $t \geq 3$, however, the similar treatment of (anti-)(monotone) $(n,k)^t$-WSAT leads to formulas with internal quantifiers that depend on k, where this dependence carries through on attempting to interchange quantifier blocks and reduce the formula. Thus the problem of $\sum_t^{\text{FO}}$ formulas that are not "nice" relates to the open problems in [**DF96**]. There appear to be other connections to the fixed-parameter hierarchies over constant-depth circuit classes defined in [**DFR96**]. Finally, we note that Cai, Chen, Downey, and Fellows [**CCDF94**] have characterized the $W[t]$ classes via a certain model of alternating log-time Turing machines. By well-known connections between these machines and constant-depth circuit classes, we feel there *should be* a closer connection between the $W[t]$ classes and uniform AC^0 than we have shown, such as might follow from removal of the "niceness" restriction in our main results. All of this points to interesting new challenges posed by the parameter k in parameterized complexity, ones that may also yield much new knowledge about "standard" complexity classes.

Acknowledgments. The third author would like to thank Drs. Paul Beame and Sam Buss for their invitation to speak at the April 1996 DIMACS Workshop on "Feasible Arithmetics and Lengths of Proofs," and some of the participants, especially Søren Riis, for relevant conversations. The main theorems were improved between Exits 141 and 130 on the Garden State Parkway on the workshop's second day.

References

[ADF93] K. Abrahamson, R. Downey, and M. Fellows, *Fixed-parameter intractability II: On completeness for W[1]*, Proc. 10th Annual Symposium on Theoretical Aspects of Computer Science, Lect. Notes in Comp. Sci., vol. 665, Springer Verlag, 1993, pp. 374–385.

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

[BFH94] H. Bodländer, M. Fellows, and M. Hallett, *Beyond NP-completeness for problems of bounded width: hardness for the W hierarchy*, Proc. 26th Annual ACM Symposium on the Theory of Computing, 1994, pp. 449–458.

[BIS90] D. Mix Barrington, N. Immerman, and H. Straubing, *On uniformity within* NC^1, J. Comp. Sys. Sci. **41** (1990), 274–306.

[Büc60] J. Büchi, *Weak second-order arithmetic and finite automata*, Zeitschrift für Mathematische Logik und Grundlagen der Mathematik **6** (1960), 66–92.

[CC93] L. Cai and J. Chen, *Fixed-parameter tractability and approximation of NP-hard optimization problems*, Proc. 2nd Israel Symp. on Theory of Computing and Systems, 1993, pp. 118–126.

[CCDF94] L. Cai, J. Chen, R. Downey, and M. Fellows, *On the structure of parametrized problems in NP*, Proc. 11th Annual Symposium on Theoretical Aspects of Computer Science, Lect. Notes in Comp. Sci., vol. 775, Springer Verlag, 1994, pp. 509–520.

[CW90] D. Coppersmith and S. Winograd, *Matrix multiplication via arithmetical progressions*, J. Symbolic Computation **9** (1990), 251–280.

[DF93] R. Downey and M. Fellows, *Fixed-parameter tractability and completeness III: Some structural aspects of the W hierarchy*, Complexity Theory: Current Research (K. Ambos-Spies, S. Homer, and U. Schöning, eds.), Cambridge Univ. Press, 1993, pp. 191–226.

[DF94] R. Downey and M. Fellows, *Parameterized computational feasibility*, Feasible Mathematics II, Birkhäuser, 1994, to appear.

[DF95] R. Downey and M. Fellows, *Fixed-parameter tractability and completeness I: Basic theory*, SIAM J. Comput. **24** (1995), 873–921.

[DF96] R. Downey and M. Fellows, *Threshold dominating sets and an improved characterization of W[2]*, 1996, Draft paper.

[DFK96] R. Downey, M. Fellows, and N. Koblitz, *Techniques for exponential parameterized reductions in vertex-set problems (incomplete draft)*, February 1996.

[DFR96] R. Downey, M. Fellows, and K. Regan, *Parameterized circuit complexity and the W hierarchy*, Theor. Comp. Sci., Ser. A (1996), to appear.

[DFT96] R. Downey, M. Fellows, and U. Taylor, *The complexity of relational database queries and an improved characterization of W[1]*, Proceedings of DMTCS'96, Springer Verlag, 1996, to appear.

[Fag74] R. Fagin, *Generalized first-order spectra and polynomial-time recognizable sets*, Complexity of Computation: Proceedings of a Symposium in Applied Mathematics of the American Mathematical Society and the Society for Industrial and Applied Mathematics, Vol. VII (R. Karp, ed.), SIAM-AMS, 1974, pp. 43–73.

[HP93] P. Hájek and P. Pudlák, *Metamathematics of first-order arithmetic*, Springer Verlag, 1993.

[HW96] M. Hallett and H..T. Wareham, *A compendium of parameterized complexity results, version 2.0*, May 1996, URL: `http://www-csc.uvic.ca/home/harold/W_hier/compendium.html`.

[Imm83] N. Immerman, *Languages which capture complexity classes*, Proc. 15th Annual ACM Symposium on the Theory of Computing, 1983, pp. 347–354.

[Imm87] N. Immerman, *Languages which capture complexity classes*, SIAM J. Comput. **16** (1987), 760–778.

[Lin92] S. Lindell, *The invariant problem for binary string structures and the parallel complexity theory of queries*, J. Comp. Sys. Sci. **44** (1992), 385–410.

[Lin94] S. Lindell, *How to define exponentiation from addition and multiplication in first-order logic on finite structures*, 1994, Unpublished manuscript. The main result will appear in a forthcoming text by N. Immerman, *Descriptive and Computational Complexity*, in the Springer Verlag "Graduate Texts in Computer Science" series.

[Lyn82] J. Lynch, *Complexity classes and theories of finite models*, Math. Sys. Thy. **15** (1982), 127–144.

[NP85] J. Nesetril and S. Poljak, *On the complexity of the subgraph problem*, Commen. Math. Univ. Carol. **26** (1985), 415–419.

[PY91] C. Papadimitriou and M. Yannakakis, *Optimization, approximation and complexity classes*, J. Comp. Sys. Sci. **43** (1991), 425–440.

[Sip83] M. Sipser, *Borel sets and circuit complexity*, Proc. 15th Annual ACM Symposium on the Theory of Computing, 1983, pp. 61–69.

[Smo91] C. Smoryński, *Logical number theory I: An introduction*, Springer Verlag, 1991.

[Sto76] L. Stockmeyer, *The polynomial time hierarchy*, Theor. Comp. Sci. **3** (1976), 1–22.

Department of Mathematics, P.O. Box 600, Victoria University, Wellington, New Zealand

E-mail address: `downey@maths.vuw.ac.nz`

Department of Computer Science, University of Victoria, Victoria, British Columbia, V8W 3P6, Canada

E-mail address: `mfellows@csr.uvic.ca`

Department of Computer Science, State University of New York at Buffalo, 226 Bell Hall, UB North Campus, Buffalo NY 14260-2000 USA

E-mail address: `regan@cs.buffalo.edu`