A Completeness Theory for Polynomial (Turing) Kernelization*

Danny Hermelin[1], Stefan Kratsch[2,**], Karolina Sołtys[3], Magnus Wahlström[4,***], and Xi Wu[5]

[1] Ben Gurion University of the Negev, Be'er Sheva, Israel
hermelin@bgu.ac.il
[2] Technical University Berlin, Germany
stefan.kratsch@tu-berlin.de
[3] Max Planck Institute for Informatics, Saarbrücken, Germany
ksoltys@mpi-inf.mpg.de
[4] Royal Holloway, University of London, England
Magnus.Wahlstrom@rhul.ac.uk
[5] University of Wisconsin Madison, Madison, USA
xiwu@cs.wisc.edu

Abstract. The framework of Bodlaender et al. (ICALP 2008, JCSS 2009) and Fortnow and Santhanam (STOC 2008, JCSS 2011) allows us to exclude the existence of polynomial kernels for a range of problems under reasonable complexity-theoretical assumptions. However, some issues are not addressed by this framework, including the existence of Turing kernels such as the "kernelization" of LEAF OUT BRANCHING(k) into a disjunction over n instances each of size poly(k). Observing that Turing kernels are preserved by polynomial parametric transformations (PPTs), we define two *kernelization hardness* hierarchies by the PPT-closure of problems that seem fundamentally unlikely to admit efficient Turing kernelizations. This gives rise to the MK- and WK-hierarchies which are akin to the M- and W-hierarchies of ordinary parameterized complexity. We find that several previously considered problems are complete for the fundamental hardness class WK[1], including MIN ONES d-SAT(k), BINARY NDTM HALTING(k), CONNECTED VERTEX COVER(k), and CLIQUE parameterized by $k \log n$. We conjecture that no WK[1]-hard problem admits a polynomial Turing kernel. Our hierarchy subsumes an earlier hierarchy of Harnik and Naor (FOCS 2006, SICOMP 2010) that, from a parameterized perspective, is restricted to classical problems parameterized by witness size. Our results provide the first natural complete problems for, e.g., their class VC_1; this had been left open.

1 Introduction

Kernelization, or data reduction, is a central concept in parameterized complexity, and has important applications outside this field as well. Roughly speaking, a kernelization

* Due to space restrictions many proofs are deferred to the full version [20].

** Main work done while supported by the Netherlands Organization for Scientific Research (NWO), project "KERNELS: Combinatorial Analysis of Data Reduction".

*** Main work done while the author was at the Max Planck Institute for Informatics.

G. Gutin and S. Szeider (Eds.): IPEC 2013, LNCS 8246, pp. 202–215, 2013.
© Springer International Publishing Switzerland 2013

algorithm reduces an instance of a given parameterized problem to an equivalent instance of size $f(k)$, where k is the parameter of the input instance. Appropriately, the function $f()$ is referred to as the *size* of the kernel. A kernel with a good size guarantee is very useful – whether one wants to solve a problem exactly, or apply heuristics, or compute an approximation, it never hurts to first apply the kernelization procedure.[1] It can also be seen more directly as instance compression, e.g., for storing a problem instance for the future use; see Harnik and Naor [18]. The common milestone for an efficient kernelization is a *polynomial kernel*, i.e., a kernel with a polynomial size guarantee. Several significant kernelization results can be found in the literature, sometimes using non-trivial mathematical tools; see, e.g., the $2k$-vertex kernel for VERTEX COVER [28], the $\mathcal{O}(k^2)$ kernel for FEEDBACK VERTEX SET [30], and the recent randomized polynomial kernel for ODD CYCLE TRANSVERSAL [26].

Fairly recently, work by Bodlaender et al. [4] together with a result of Fortnow and Santhanam [17] provided the first technique to rule out the existence of any polynomial kernel for certain problems, assuming that NP $\not\subseteq$ coNP/poly (and PH does not collapse [31]). A series of further papers have applied this framework to concrete problems and developed it further, e.g., [13,12,5,11,21,22,14].

However, there are relaxed notions of efficient kernelization which are not ruled out by any existing work, but which would still be useful in practice and interesting from a theoretical point of view. Almost immediately after the appearance of the above lower bounds framework, the question was raised whether there were notions of "cheating" kernels. For example for the problem k-PATH which could circumvent the above lower bounds by producing Turing kernels instead of standard many-one kernelizations [3]. Not long afterwards, the first example of such a cheating kernel appeared: Binkele-Raible et al. [2] showed that the k-LEAF OUT-BRANCHING problem (given a directed graph G and an integer k, does G contain a directed tree with at least k leaves?) does not admit a polynomial kernel unless NP $\not\subseteq$ coNP/poly, but does admit one (with $\mathcal{O}(k^3)$ vertices) if the root of the tree is fixed, implying a Turing kernel in the form of a disjunction over n instances, each of size polynomial in k. There are also simpler problems sharing the same behavior; for example, the problem of CLIQUE parameterized by maximum degree is trivially compatible with the lower-bound frameworks, implying that it has no polynomial many-one kernel unless NP $\not\subseteq$ coNP/poly, but admits a very simple Turing kernel into n instances of k vertices (by taking the neighborhood of each vertex). We will call such a disjunctive Turing kernel an *OR-kernel*. We observe that many of the positive aspects of standard (many-one) polynomial kernels are preserved by OR-kernels, or even generally Turing kernels; in particular the algorithmic consequences (e.g., a Turing kernel with polynomial individual instance sizes for a problem in NP implies an algorithm with a running time of $2^{k^{\mathcal{O}(1)}}n^{\mathcal{O}(1)}$, same as for a polynomial many-one kernel).

The question of the extent to which such Turing kernels exist is theoretically very interesting and one of the most important problems in the field. Some restricted forms of Turing kernels, e.g., polynomial AND-kernels, can already be excluded by the existing framework, as they are special cases of polynomial kernels which may use conondeterministic polynomial time (cf. [12,22,26]). However, for OR-kernels or Turing

[1] This is assuming that the kernel preserves solution values, which most do.

kernels the current framework does not apply (as also witnessed by the k-LEAF OUT-BRANCHING and CLIQUE by max degree problems). It is also unclear if the framework could be adapted to deal with them. Instead, we take an approach common in complexity theory, namely that of defining an appropriate notion of hardness, and studying problems that are complete under this notion. We start from a set of problems for which we conjecture that none of them has a polynomial Turing kernel, and show that they are equivalent under PPT-reductions (which preserve existence of polynomial Turing kernels). The result is a robust class of hardness of Turing kernelization, dubbed WK[1], whose complete problems include central problems from different areas of theoretical computer science. While we have no concrete evidence that our conjecture holds, we feel that the abundance of WK[1]-complete problems, where Turing-kernelization is found for none of them, might suggest its validity.

WK[1]-hard problems. A cornerstone problem of WK[1] is the k-step halting problem for non-deterministic Turing machines parameterized by $k \log n$. To see why this is a powerful problem, and why an efficient Turing kernel would seem unlikely, consider the k-clique problem. Given a graph G with n vertices and an integer k, it is easy to construct a Turing machine which checks in poly(k) non-deterministic steps whether G contains a k-clique (by using a number of states polynomial in n). On the other hand, an OR-kernel for the problem (or more generally, a polynomial Turing kernel) would require reducing k-clique to a polynomial number of questions of size polynomial in $k \log n$ (e.g., poly(n) induced subgraphs of G, each of size poly$(k \log n)$, which are guaranteed to cover any k-clique of G). In fact, the above-mentioned halting problem captures not only clique, but all problems where a witness of t bits can be verified in poly$(t, \log n)$ time (e.g., SUBGRAPH ISOMORPHISM).

Other WK[1]-complete problems include MIN ONES d-SAT, the problem of finding a satisfying assignment with at most k true variables for a d-CNF formula, parameterized by the solution size k; HITTING SET parameterized by the number of sets or hyperedges m; and CONNECTED VERTEX COVER parameterized by the solution size k. Of these, we in particular want to single out MIN ONES d-SAT, which captures all minimization problems for which the consistency of a solution can be locally verified (by looking at combinations of d values at a time). For example, this includes the $\mathcal{H}$-FREE EDGE MODIFICATION(k) problems, where $\mathcal{H}$ is a finite, fixed set of forbidden induced subgraphs, and the goal is to remove or add k edges in the input graph in order to obtain a graph with no induced subgraph in $\mathcal{H}$ [8].

Extending the hardness class WK[1], we also define a hierarchy of hardness classes WK[t] and MK[t] for $t \geq 1$, mirroring the W- and M-hierarchies of traditional parameterized complexity; see [16]. We note that there are also strong similarities to the work of Harnik and Naor [18], in particular to the VC-hierarchy (which is defined around the notion of *witness length* for problems in NP). However, the notion of a parameter seems more general and robust than witness length; consider for example the volume of work in FPT on structural parameters such as treewidth. We also feel that the connections to the traditional FPT hardness classes (see Section 3) flesh out and put into context Harnik and Naor's work, and the link to the Turing kernel question adds interest to the separation question. Still, the main focus of our work is the hardness class WK[1].

We hope that our conjecture, that WK[1] does not have polynomial Turing kernels, will inspire other researchers to revisit the kernelization properties of problems which have been shown not to admit standard polynomial kernels unless PH collapses, but for which hardness for the above-mentioned class is less obvious. In particular, we leave open the WK[1]-hardness of k-PATH, the problem for which the existence of Turing kernels was originally asked in [3].

2 Preliminaries

We begin our discussion by formally defining some of the main concepts used in this paper, and by introducing some terminology and notation that will be used throughout. All problem definitions are deferred to the full version of this paper [20]. We use $[n]$ to denote the set of integers $\{1, \ldots, n\}$.

Definition 1 (Kernelization). *A* kernelization algorithm*, or, in short, a* kernel *for a parameterized problem $L \subseteq \Sigma^* \times \mathbb{N}$ is a polynomial-time algorithm that on a given input $(x, k) \in \Sigma^* \times \mathbb{N}$ outputs a pair $(x', k') \in \Sigma^* \times \mathbb{N}$ such that $(x, k) \in L \Leftrightarrow (x', k') \in L$, and $|x'| + k' \leq f(k)$ for some function f. The function f above is referred to as the* size *of the kernel.*

In other words, a kernel is a polynomial-time reduction from a problem to itself that compresses the problem instance to a size depending only on the parameter. If the size of a kernel for L is polynomial, we say that L has a *polynomial kernel*. In the interest of robustness and ease of presentation, we relax the notion of kernelization to allow the output to be an instance of a different problem. This has been referred to as a generalized kernelization [4] or bikernelization [1]. The class of all parameterized problems with polynomial kernels in this relaxed sense is denoted by PK.

Definition 2 (Turing Kernelization). *A* Turing kernelization *for a parameterized problem $L \subseteq \Sigma^* \times \mathbb{N}$ is a polynomial-time algorithm with oracle access to a parameterized problem L' that can decide whether an input (x, k) is in L using queries of size bounded by $f(k)$, for some computable function f. The function f is referred to as the* size *of the kernel.*

If the size is polynomial, we say that L has a *polynomial Turing kernel*. The class of all parameterized problems with polynomial Turing kernels is denoted by Turing-PK.

Definition 3 (Polynomial Parametric Transformations [7]). *Let L_1 and L_2 be two parameterized problems. We write $L_1 \leq_{ppt} L_2$ if there exists a polynomial time computable function $\Psi : \{0,1\}^* \times \mathbb{N} \to \{0,1\}^* \times \mathbb{N}$ and a constant $c \in \mathbb{N}$, such that for all $(x, k) \in \Sigma^* \times \mathbb{N}$, if $(x', k') = \Psi(x, k)$ then $(x, k) \in L_1 \iff (x', k') \in L_2$, and $k' \leq ck^c$. The function Ψ is called a* polynomial parameter transformation *(*PPT *for short). If $L_1 \leq_{ppt} L_2$ and $L_2 \leq_{ppt} L_1$ we write $L_1 \equiv_{ppt} L_2$.*

Proposition 1. *Let L_1, L_2, and L_3 be three parameterized problems.*

- *If $L_1 \leq_{ppt} L_2$ and $L_2 \leq_{ppt} L_3$ then $L_1 \leq_{ppt} L_3$.*
- *If $L_1 \leq_{ppt} L_2$ and $L_2 \in$* PK *(resp.* Turing-PK*) then $L_1 \in$* PK *(resp.* Turing-PK*).*

We denote parameterizations in parentheses after the problem name, for example, CLIQUE$(k \log n)$. In this example, k is the solution size, and n the size of the input. (Recall that CLIQUE(k) is one of the fundamental hard problems for parameterized complexity, and unlikely to admit a kernel of any size [16]; however, under a parameter $p = k \log n$ it has a trivial kernel of size 2^p.)

Note that, if a problem Q is solvable in $2^{k^{\mathcal{O}(1)}} n^{\mathcal{O}(1)}$ time, then $\mathrm{Q}(k) \equiv_{ppt} \mathrm{Q}(k \log n)$.

3 The WK- and MK-Hierarchies

In this section we introduce our hierarchies of inefficient kernelizability, the MK- and WK-hierarchies. Relations to the so-called *VC-hierarchy* of Harnik and Naor [18] are discussed in Section 3.1. To begin with, for $t \geq 0$ and $d \geq 1$, we inductively define the following classes $\Gamma_{t,d}$ and $\Delta_{t,d}$ of formulas following [16]:

$$\begin{aligned}
\Gamma_{0,d} &:= \{\lambda_1 \wedge \cdots \wedge \lambda_c : c \in [d] \text{ and } \lambda_1, \ldots, \lambda_c \text{ are literals}\},\\
\Delta_{0,d} &:= \{\lambda_1 \vee \cdots \vee \lambda_c : c \in [d] \text{ and } \lambda_1, \ldots, \lambda_c \text{ are literals}\},\\
\Gamma_{t+1,d} &:= \{\textstyle\bigwedge_{i \in I} \delta_i : I \text{ is a finite non-empty set and } \delta_i \in \Delta_{t,d} \text{ for all } i \in I\},\\
\Delta_{t+1,d} &:= \{\textstyle\bigvee_{i \in I} \gamma_i : I \text{ is a finite non-empty set and } \gamma_i \in \Gamma_{t,d} \text{ for all } i \in I\}.
\end{aligned}$$

Thus, $\Gamma_{1,3}$ is the set of all 3-CNF formulas, and $\Gamma_{2,1}$ is the set of all CNF formulas. Given a class Φ of propositional formulas, we let $\Phi^+, \Phi^- \subseteq \Phi$ denote the restrictions of Φ to formulas containing only positive and negative literals, respectively. For any given Φ, we define two parameterized problems:

- Φ-WSAT$(k \log n)$ is the problem of determining whether a formula $\phi \in \Phi$ with n variables has a satisfying assignment of Hamming weight exactly k, parameterized by $k \log n$.
- Φ-SAT(n) is the problem of determining whether a formula $\phi \in \Phi$ with n variables is satisfiable, parameterized by n.

In particular, we will be interested in $\Gamma_{t,d}$-WSAT$(k \log n)$ and $\Gamma_{t,d}$-SAT(n).

We now reach our class definitions. For a parameterized problem $L \subseteq \Sigma^* \times \mathbb{N}$, we let $[L]_{\leq_{ppt}}$ denote the closure of L under polynomial parametric transformations. That is, $[L]_{\leq_{ppt}} := \{L' \subseteq \Sigma^* \times \mathbb{N} : L' \leq_{ppt} L\}$.

Definition 4. *Let $t \geq 1$ be an integer. The classes* WK$[t]$ *and* MK$[t]$ *are defined by*

- $\mathrm{WK}[t] := \bigcup_{d \in \mathbb{N}} [\Gamma_{t,d}\text{-WSAT}(k \log n)]_{\leq_{ppt}}$.
- $\mathrm{MK}[t] := \bigcup_{d \in \mathbb{N}} [\Gamma_{t,d}\text{-SAT}(n)]_{\leq_{ppt}}$.

The naming of the classes in our hierarchies comes from the close relationship of the MK- and WK-hierarchies to the M- and W-hierarchies of traditional parameterized complexity [16]. Roughly speaking, WK$[t]$ and MK$[t]$ are reparameterizations by a factor of $\log n$ (or log of the instance size) of the traditional parameterized complexity classes W$[t]$ and M$[t]$ (although W$[t]$ and M$[t]$ are closed under FPT reductions, which may use superpolynomial time in k).

There are also close connections to the so-called *subexponential time* S-hierarchy (see [16, Chapter 16]); specifically, S$[t]$ and MK$[t]$ are defined from the same starting

problems, using closures under different types of reduction; see also Cygan et al. [10]. We further note that [10] asked as an open problem, a reduction which in our terms would go from an MK[2]-complete problem to one in WK[1], and our work suggests the difficulty of producing one (see Theorem 2).

We show the following complete problems for our hierarchy.

Theorem 1 (★[2]). *Let $t \geq 1$. The following hold.*

- $\Gamma^{-}_{1,2}$-WSAT$(k \log n)$ *is* WK$[1]$*-complete.*
- $\Gamma^{-}_{t,1}$-WSAT$(k \log n)$ *is* WK$[t]$*-complete for odd $t > 1$.*
- $\Gamma^{+}_{t,1}$-WSAT$(k \log n)$ *is* WK$[t]$*-complete for even $t > 1$.*
- $\Gamma_{1,d}$-SAT(n) *is* MK$[1]$*-complete for every $d \geq 3$.*
- $\Gamma_{t,1}$-SAT(n) *is* MK$[t]$*-complete for $t \geq 2$.*

Theorem 1 above shows that the traditional problems used for showing completeness in the W- and M-hierarchies have reparameterized counterparts which are complete for our hierarchy. The theorem is proven using a set of PPTs from the specific class-defining problems to the corresponding target problem in the theorem. Our main contribution is a PPT for the first item, for which previous proofs used FPT-time reductions. The remaining items are either easy or well-known.

We now proceed to show the class containments in our hierarchy. The main containments are as follows.

Theorem 2 (★). MK$[1] \subseteq$ WK$[1] \subseteq$ MK$[2] \subseteq$ WK$[2] \subseteq$ MK$[3] \subseteq \cdots \subseteq$ EXPT.

We also study a few further particular classes. First, let PK_{NP} denote the class of parameterized problems with polynomial kernels whose output problem lies in NP. We have the following relationship.

Lemma 1 (★). MK$[1] = \text{PK}_{\text{NP}}$.

Proof (sketch). If a problem has a polynomial kernel within NP, then we may first kernelize, then reduce to 3-SAT by the NP-completeness of the latter. Conversely, any problem in MK[1] has a PPT to d-SAT(n) for some constant d, which (as the latter has bounded size) forms a kernelization within NP. □

Next, for a problem L, we define a parameterized problem OR$(L)(\ell)$ where the input is a set of instances $x_1, \ldots, x_t$ of L, each of length at most ℓ, and the task is to decide whether $x_i \in L$ for at least one instance x_i.

Lemma 2 (★). *Let L be an* NP*-complete language. Then* MK$[1] \subseteq [\text{OR}(L)(\ell)]_{\leq_{ppt}} \subseteq$ WK$[1]$*, where the first inclusion is strict unless* NP $\subseteq$ coNP/poly*, and the second is strict unless every problem in* WK$[1]$ *has a polynomial OR-kernel.*

Proof (sketch). The first containment is trivial; the second can be given via reduction to the BINARY NDTM HALTING problem (see Section 4). The first consequence follows from Fortnow and Santhanam [17]; the second follows since OR(L) has an OR-kernel, and OR-kernels are preserved by PPTs. □

[2] The proofs for lemmas and theorems denoted by a star are deferred to the full version [20].

For AND(L), and problems with Turing kernels more generally, no similar containment is known. However, our hierarchy can still be useful for AND-compositional problems, in showing them to be *hard* for some level.

3.1 Comparison with the VC-Hierarchy

We now discuss the relations between the VC-hierarchy of Harnik and Naor [18] and the MK- and WK-hierarchies defined in this paper. Let us review some definitions. An NP-*language* L is for the purposes of this section defined by a pair (R_L, k), where $R_L(\cdot,\cdot)$ is a polynomial-time computable relation and $k(x) = |x|^{\mathcal{O}(1)}$ a polynomial-time computable function, and $x \in L$ for an instance x if and only if there is some string y with $|y| \leq k(x)$, such that $R_L(x, y)$ holds. The string y is called the *witness* for x. This naturally defines a parameterization of L, with parameter $k(x)$; the resulting problem is FPT (with running time $O^*(2^k)$). We refer to this parameterized problem as the *direct parameterization* of (R_L, k). Harnik and Naor consider the feasibility of a (possibly probabilistic) *compression* of an instance x of L into length poly$(k(x))$, in a sense essentially equivalent to our (relaxed) notion of kernelization; see [18] for details.

Before we give the technical results, let us raise two points. First, Harnik and Naor deal solely with problems where the parameter is the length $k(x)$ of a witness. Although this always defines a valid parameter (as we have seen), it is not clear that every reasonable parameterization of an NP-problem can be interpreted as a witness in this sense. Second, although Harnik and Naor are rather lax with the choice of witness (frequently letting it go undefined), we want to stress that the choice of witness can have a big impact on the kernelization complexity of a problem. Consider as an example the case of HITTING SET (treated later in this paper). Let n denote the number of vertices of an instance, m the number of edges, and k the upper bound on solution size; note that the total coding size is $\mathcal{O}(mn)$. We will find that the problem is WK[1]-complete when parameterized by m, MK[2]-complete under the parameter n, and WK[2]-complete under the parameter $k \log n$. The two latter both represent plausible choices of witnesses; a witness of length m is less obvious (or natural), but there is a simple witness of length $m \log k \leq m \log m$ obtained by describing a partition of the edges into k sets. One may also consider structural parameters such as treewidth (e.g., of the bipartite vertex/edge incidence graph), for which a corresponding short witness seems highly unlikely (but which can be shown to be MK[2]-hard). That said, it is not hard to see that every problem in the WK- and MK-hierarchies has a corresponding witness, by the PPT-reduction that proves its class membership.

For reasons of space, we give only a brief summary of the results; see the full version [20] for more details.

Theorem 3 (★). *Let L be an* NP*-language defined by (R_L, k), and let $\mathcal{Q}$ be its direct parameterization. The following hold.*

1. *L is contained in (hard for, complete for) $\mathcal{VC}_{\text{or}}$ if and only if $\mathcal{Q}$ is contained in (hard for, complete for) the PPT-closure of* OR(3-SAT).
2. *L is contained in $\mathcal{VC}_1$ ($\mathcal{VC}_1$-hard, $\mathcal{VC}_1$-complete) if and only if $\mathcal{Q}$ is contained in* WK[1] *(*WK[1]*-hard,* WK[1]*-complete).*

3. *L is contained in $\mathcal{VC}_t$ ($\mathcal{VC}_t$-hard, $\mathcal{VC}_t$-complete) for $t > 1$ if and only if $\mathcal{Q}$ is contained in* MK$[t]$ *(*MK$[t]$*-hard,* MK$[t]$*-complete).*

Thus, we answer a question left open in [18], of finding a natural problem complete for $\mathcal{VC}_1$ (as well as some minor questions about specific problem placements).

4 Complete Problems for WK[1]

In this section we show that several natural problems are complete for our fundamental hardness class WK$[1]$ and thus exemplify its robustness. Our starting point will be CLIQUE$(k \log n)$ which is clearly equivalent to $\Gamma_{1,2}^{-}$-WSAT$(k \log n)$; the latter is WK$[1]$-complete by Theorem 1.

Theorem 4. CLIQUE$(k \log n)$ *is complete for* WK$[1]$.

4.1 Basic Problems

This section establishes the following theorem that covers some basic problems which will be convenient for showing WK[1]-hardness and completeness for other problems. Standard many-one polynomial kernels for these problems were excluded in previous work [4,24,25,13].

Theorem 5. *The following problems are all complete for* WK$[1]$*:*

- BINARY NDTM HALTING(k) *and* NDTM HALTING$(k \log n)$.
- MIN ONES d-SAT(k) *for $d \geq 3$, with at most k true variables.*
- HITTING SET(m) *and* EXACT HITTING SET(m)*, with m sets.*
- SET COVER(n) *and* EXACT SET COVER(n)*, with n elements.*

The following colorful variants are helpful for our reductions.

Lemma 3 ([15,13]). *The following equivalences hold.*

- MULTICOLORED CLIQUE$(k \log n) \equiv_{ppt}$ CLIQUE$(k \log n)$.
- MULTICOLORED HITTING SET$(m) \equiv_{ppt}$ HITTING SET(m).

We now proceed with the reductions. For many problems, we will find it convenient to show hardness by reduction from EXACT HITTING SET(m) or HITTING SET(m); hence we begin by showing the completeness of these problems. We give a chain of reductions from CLIQUE$(k \log n)$, via EXACT HITTING SET(m) and NDTM HALTING$(k \log n)$, and back to CLIQUE$(k \log n)$, closing the cycle. After this we will treat the HITTING SET(m) problem. Note that NDTM HALTING$(k \log n)$ and BINARY NDTM HALTING(k) (the problem restricted to machines with a binary tape alphabet) are easily PPT-equivalent.

Lemma 4 (★). MULTICOLORED CLIQUE$(k \log n) \leq_{ppt}$ EXACT HITTING SET(m).

Proof (sketch). Let the input be a graph $G = (V, E)$ with coloring function $c : V \to [k]$. Assume $V = [n]$, and let $b_\ell(v)$ be the ℓ:th bit in the binary expansion of v. We create a set family $\mathcal{F} \subseteq 2^E$ with sets $F_{i,j} = \{uv \in E : c(u) = i, c(v) = j\}$ for $1 \leq i < j \leq k$, and $F_{i,j,j',\ell} = \{uv \in E : c(u) = i, c(v) = j, b_\ell(u) = 1\} \cup \{uv \in E : c(u) = i, c(v) = j', b_\ell(u) = 0\}$ for all color pairs (i, j) and (i, j') (e.g., for $i, j, j' \in [k]$ with $i \neq j, j'$ and $j < j'$), $1 \leq \ell \leq \lceil \log n \rceil$. By the first set family, exactly one edge per color class is selected in a solution; the second family ensures that the selections are consistent (i.e., for each color class i, all incident edges are incident on the same vertex u). □

Lemma 5 (★). EXACT HITTING SET$(m) \leq_{ppt}$ NDTM HALTING$(k \log n)$.

Proof (sketch). Let $\mathcal{F} = \{F_1, \ldots, F_m\}$ be a set family on universe $U = [n]$; we will create a NDTM with tape alphabet $[n]$ which verifies whether $\mathcal{F}$ has an exact hitting set. We may assume $n \leq 2^m$. In the first phase, for each $i \in [m]$ we put the ID of a member u_i of F_i in cell i; in subsequent phases, we verify that no set F_i is hit twice (e.g., if $u_j \neq u_i$, then $u_j \notin F_i$). This is easily done in poly(m) steps, by a machine with poly$(n + m)$ states; thus $k \log n = m^{O(1)}$ and we are done. □

Lemma 6 (★). BINARY NDTM HALTING$(k) \leq_{ppt}$ MIN ONES 3-SAT(k).

Proof (sketch). Let M be a Turing machine with ℓ transitions in the state diagram. We will create a 3-CNF formula ϕ that has a satisfying assignment of Hamming weight at most k', $k' = k^{O(1)}$, if and only if M accepts within k steps. We use variables $M_{e,t}, e \in [\ell], t \in [k]$, designating that M uses transition e of the state diagram as the t:th execution step, along with auxiliary variables tracing the machine state, head position, and tape contents. By using the log-cost selection formulas of [25], we can ensure that exactly one variable $M_{e,t}$ is true for each t, at the cost of an extra solution weight of $k \log \ell$. Given this, the consistency of an assignment can be enforced using only local (3-ary) conditions (details omitted). We also require that the final state is accepting. With some care, one can ensure that every satisfying assignment has the same weight $k' =$ poly$(k, \log \ell)$. Since the problem is FPT, we may assume $\log \ell \leq k$, thus we have polynomial reduction time and parameter growth, i.e., a PPT. □

The following lemma is a direct consequence of Theorem 1 in Section 3.

Lemma 7 (★). MIN ONES d-SAT$(k) \leq_{ppt}$ CLIQUE$(k \log n)$ *for every fixed* d.

The remaining problems in Theorem 5 for which we need to show completeness are HITTING SET(m), SET COVER(n), and EXACT SET COVER(n). Since it is well known that HITTING SET$(m) \equiv_{ppt}$ SET COVER(n) and EXACT HITTING SET$(m) \equiv_{ppt}$ EXACT SET COVER(n), we finish the proof of Theorem 5 by proving WK[1]-completeness for HITTING SET(m).

Lemma 8 (★). HITTING SET(m) *is* WK[1]*-complete.*

4.2 Further Problems

We briefly list further problems which we have shown to be WK[1]-complete or WK[1]-hard. LOCAL CIRCUIT SAT$(k \log n)$ was defined by Harnik and Naor, for defining their VC-hierarchy [18]. The remaining proofs are adaptations of lower bounds proofs by Bodlaender *et al.* [7] and Dom *et al.* [13].

Theorem 6 (★). *The following problems are* WK[1]*-complete.*

- LOCAL CIRCUIT SAT$(k \log n)$.
- MULTICOLORED PATH(k) *and* DIRECTED MULTICOLORED PATH(k).
- MULTICOLORED CYCLE(k) *and* DIRECTED MULTICOLORED CYCLE(k).
- CONNECTED VERTEX COVER(k).
- CAPACITATED VERTEX COVER(k).
- STEINER TREE$(k+t)$*, for solution size* k *and* t *terminals.*
- SMALL SUBSET SUM(k) *(see [13] for parameter definition).*
- UNIQUE COVERAGE(k)*, where* k *is the number of items to be covered.*

Theorem 7 (★). *The following problems are* WK[1]*-hard.*

- DISJOINT PATHS(k) *and* DISJOINT CYCLES(k).

5 Problems in Higher Levels

In this section we investigate the second level of the MK- and WK-hierarchies, and present some complete and hard problems for these classes.

MK[2]-complete problems. According to Theorem 1, MK[2] is the PPT-closure of the classical CNF satisfiability problem where the parameter is taken to be the number of variables in the input formula. The PPT-equivalence of this problem to HITTING SET(n) and SET COVER(m) is well known.

Theorem 8. HITTING SET(n) *and* SET COVER(m) *are complete for* MK[2].

Heggernes et al. [19] consider RESTRICTED PERFECT DELETION$(|X|)$ and RESTRICTED WEAKLY CHORDAL DELETION$(|X|)$, where the input is a graph G, a set X of vertices of G such that $G - X$ is perfect (resp. weakly chordal), and an integer k, and the task is to select at most k vertices $S \subseteq X$ such that $G - S$ is perfect (resp. weakly chordal). We get the following corollary from Theorem 8 and PPTs given in [19].

Corollary 1. RESTRICTED PERFECT DELETION$(|X|)$ *and* RESTRICTED WEAKLY CHORDAL DELETION$(|X|)$ *are hard for* MK[2].

WK[2]-complete problems. Due to space limitations we only state the following completeness and hardness results for WK[2] and defer the proofs to the full version.

Theorem 9 (★). *The following problems are complete for* WK[2]*:*

- HITTING SET$(k \log n)$ *and* SET COVER$(k \log m)$.
- DOMINATING SET$(k \log n)$ *and* INDEPENDENT DOMINATING SET$(k \log n)$.
- STEINER TREE$(k \log n)$

From Theorem 9, we immediately get the following corollary via PPTs by Lokshtanov [27] and Heggernes *et al.* [19].

Corollary 2. *The following problems are all hard for* WK[2]*:*

- WHEEL-FREE DELETION$(k \log n)$.
- PERFECT DELETION$(k \log n)$.
- WEAKLY CHORDAL DELETION$(k \log n)$.

For the first four problems in Theorem 9 the results follow easily (see [20]), so let us focus on the more interesting case of STEINER TREE$(k \log n)$. While WK[2]-hardness for this problem follows immediately from *e.g.* the PPT from HITTING SET$(k \log n)$ given in [13], showing membership in WK[2] is more challenging. To facilitate this and other non-trivial membership proofs, we consider the issue of a machine characterization of WK[2], similarly to the WK[1]-complete NDTM HALTING$(k \log n)$ problem. The natural candidate would be MULTI-TAPE NDTM HALTING$(k \log n)$, as this same problem with parameter k is W[2]-complete [9]. However, while the problem with parameter $k \log n$ is easily shown to be WK[2]-hard, we were so far unable to show WK[2]-membership. On the other hand, the following extension of a single-tape non-deterministic Turing machine leads to a WK[2]-complete problem, which we name NDTM HALTING WITH FLAGS.

Definition 5. *A* (single-tape, non-deterministic) Turing machine with flags *is a standard (single-tape, non-deterministic) Turing machine which in addition to its working tape has access to a set F of* flags*. Each state transition of the Turing machine has the ability to read and/or write a subset of the flags. A transition that* reads *a set $S \subseteq F$ of flags is only applicable if all flags in S are set. A transition that* writes *a set $S \subseteq F$ of flags causes every flag in S to be set. In the initial state, all flags are unset. Note that there is no operation to reset a flag.*

Theorem 10. NDTM HALTING WITH FLAGS$(k \log n)$ *is* WK[2]*-complete.*

Proof. Showing WK[2]-hardness is easy by reduction from HITTING SET$(k \log n)$. In fact, the hitting set instance can be coded directly into the flags, without any motion of the tape head – simply construct a machine that non-deterministically makes k non-writing transitions, each corresponding to including a vertex in the hitting set, followed by one verification step. The machine has m flags, one for every set in the instance, and a step corresponding to selecting a vertex v activates all flags corresponding to sets containing v. Finally, the step to the accepting state may only be taken if all flags are set. By assuming $\log m \leq k \log n$ (or else solving the instance exactly) we get a PPT.

Showing membership in WK[2] can be done by translation to $\Gamma_{2,1}$-WSAT$(k \log n)$. The transition is similar to that in Lemma 6. The only complication is to enforce consistency of transitions which read and write sets of flags, but this is easily handled. Let $M_{e,t}$ signify that step number t of the machine follows edge e of the state diagram (as in Lemma 6). If transition e has a flag f as a precondition, then we simply add a clause

$$(\neg M_{e,t} \vee M_{e_{i_1},1} \vee \ldots \vee M_{e_{i_m},1} \vee \ldots \vee M_{e_{i_1},t-1} \vee \ldots \vee M_{e_{i_m},t-1}),$$

where $e_{i_1}, \ldots, e_{i_m}$ is an enumeration of all transitions in the state diagram which set flag f. The rest of the reduction proceeds without difficulty. □

Lemma 9 (★). STEINER TREE$(k \log n) \leq_{ppt}$ NDTM HALTING WITH FLAGS$(k \log n)$.

6 Discussion

We have defined a hierarchy of PPT-closed classes, akin to the M- and W-hierarchy of parameterized intractability, in order to build up a completeness theory for polynomial (Turing) kernelization. The fundamental hardness class is called WK$[1]$ and we conjecture that no WK$[1]$-hard problem admits a polynomial Turing kernelization. At present, the state of the art in lower bounds for kernelization does not seem to provide a way to connect this conjecture to standard complexity assumptions. However, there is collective evidence by a wealth of natural problems that are complete for WK$[1]$ and for which polynomial Turing kernels seem unlikely. (Recall that admittance of Turing kernels is preserved by PPTs and hence a single polynomial Turing kernel would transfer to all WK$[1]$ problems.) Of course, our examples provide only a partial image of the WK[1] landscape. For example, the various kernelizability dichotomies that have been shown for CSP problems [25,23] can be shown to imply dichotomies between problems with polynomial kernels and WK[1]-complete problems (and in some cases the third class of W[1]-hard problems). We take this as further evidence of the naturalness of the class.

On the more structural side, we have discussed the relation to the earlier VC-hierarchy of Harnik and Naor [18] which, from our perspective, is restricted to NP-problems parameterized by witness size. Under this interpretation their hierarchy folds into ours, with the levels of their hierarchy mapping to a subset of the levels of our hierarchy.

Many questions remain. One is the WK[1]-hardness of PATH(k) and CYCLE(k); for these problems, we have only lower bound proofs in the framework of Bodlaender *et al.* [4], leaving the question of Turing kernels open. There are also several problems, including the work on structural graph parameters by Bodlaender, Jansen, and Kratsch, *e.g.* [6], which we have not investigated. It is also unknown whether MULTI-TAPE NDTM HALTING$(k \log n)$ is in WK[2]. Furthermore, it would be interesting to know some natural parameterized problems which are WK[2]-complete under a standard parameter (e.g., k rather than $k \log n$).

Still, the main open problem is to provide (classical or parameterized) complexity theoretical implications of polynomial Turing kernelizations for WK[1]. More modest variants of this goal include excluding only OR-kernels, and/or considering the general problem of Turing machine acceptance parameterized by witness length.

References

1. Alon, N., Gutin, G., Kim, E.J., Szeider, S., Yeo, A.: Solving max-*r*-sat above a tight lower bound. Algorithmica 61(3), 638–655 (2011)
2. Binkele-Raible, D., Fernau, H., Fomin, F.V., Lokshtanov, D., Saurabh, S., Villanger, Y.: Kernel(s) for problems with no kernel: On out-trees with many leaves. ACM Transactions on Algorithms 8(4), 38 (2012)
3. Bodlaender, H.L., Demaine, E.D., Fellows, M.R., Guo, J., Hermelin, D., Lokshtanov, D., Müller, M., Raman, V., van Rooij, J., Rosamond, F.A.: Open problems in parameterized and exact computation – IWPEC 2008. Technical Report UU-CS-2008-017, Dept. of Informatics and Computing Sciences, Utrecht University (2008)
4. Bodlaender, H.L., Downey, R.G., Fellows, M.R., Hermelin, D.: On problems without polynomial kernels. Journal of Computer and System Sciences 75(8), 423–434 (2009)

5. Bodlaender, H.L., Jansen, B.M.P., Kratsch, S.: Cross-composition: A new technique for kernelization lower bounds. In: Proc. of the 28th international Symposium on Theoretical Aspects of Computer Science (STACS), pp. 165–176 (2011)
6. Bodlaender, H.L., Jansen, B.M.P., Kratsch, S.: Kernel bounds for path and cycle problems. In: Marx, D., Rossmanith, P. (eds.) IPEC 2011. LNCS, vol. 7112, pp. 145–158. Springer, Heidelberg (2012)
7. Bodlaender, H.L., Thomassé, S., Yeo, A.: Kernel bounds for disjoint cycles and disjoint paths. Theor. Comput. Sci. 412(35), 4570–4578 (2011)
8. Cai, L.: Fixed-parameter tractability of graph modification problems for hereditary properties. Information Processing Letters 58(4), 171–176 (1996)
9. Cesati, M.: The Turing way to parameterized complexity. J. Comput. Syst. Sci. 67(4), 654–685 (2003)
10. Cygan, M., Dell, H., Lokshtanov, D., Marx, D., Nederlof, J., Okamoto, Y., Paturi, R., Saurabh, S., Wahlström, M.: On problems as hard as CNF-SAT. In: IEEE Conference on Computational Complexity, pp. 74–84 (2012)
11. Dell, H., Marx, D.: Kernelization of packing problems. In: Rabani [29], pp. 68–81
12. Dell, H., van Melkebeek, D.: Satisfiability allows no nontrivial sparsification unless the polynomial-time hierarchy collapses. In: Proc. of the 42th Annual ACM Symposium on Theory of Computing (STOC), pp. 251–260 (2010)
13. Dom, M., Lokshtanov, D., Saurabh, S.: Incompressibility through colors and IDs. In: Albers, S., Marchetti-Spaccamela, A., Matias, Y., Nikoletseas, S., Thomas, W. (eds.) ICALP 2009, Part I. LNCS, vol. 5555, pp. 378–389. Springer, Heidelberg (2009)
14. Drucker, A.: New limits to classical and quantum instance compression. In: FOCS, pp. 609–618. IEEE Computer Society (2012)
15. Fellows, M.R., Hermelin, D., Rosamond, F.A., Vialette, S.: On the parameterized complexity of multiple-interval graph problems. Theoretical Computer Science 410(1), 53–61 (2009)
16. Flum, J., Grohe, M.: Parameterized Complexity Theory. Springer-Verlag New York, Inc. (2006)
17. Fortnow, L., Santhanam, R.: Infeasibility of instance compression and succinct PCPs for NP. J. Comput. Syst. Sci. 77(1), 91–106 (2011)
18. Harnik, D., Naor, M.: On the compressibility of NP instances and cryptographic applications. SIAM Journal on Computing 39(5), 1667–1713 (2010)
19. Heggernes, P., van't Hof, P., Jansen, B.M.P., Kratsch, S., Villanger, Y.: Parameterized complexity of vertex deletion into perfect graph classes. In: Owe, O., Steffen, M., Telle, J.A. (eds.) FCT 2011. LNCS, vol. 6914, pp. 240–251. Springer, Heidelberg (2011)
20. Hermelin, D., Kratsch, S., Soltys, K., Wahlström, M., Wu, X.: Hierarchies of inefficient kernelizability. CoRR, abs/1110.0976 (2011)
21. Hermelin, D., Wu, X.: Weak compositions and their applications to polynomial lower bounds for kernelization. In: Rabani [29], pp. 104–113
22. Kratsch, S.: Co-nondeterminism in compositions: A kernelization lower bound for a Ramsey-type problem. In: Rabani [29], pp. 114–122
23. Kratsch, S., Marx, D., Wahlström, M.: Parameterized complexity and kernelizability of max ones and exact ones problems. In: Hliněný, P., Kučera, A. (eds.) MFCS 2010. LNCS, vol. 6281, pp. 489–500. Springer, Heidelberg (2010)
24. Kratsch, S., Wahlström, M.: Two edge modification problems without polynomial kernels. In: Chen, J., Fomin, F.V. (eds.) IWPEC 2009. LNCS, vol. 5917, pp. 264–275. Springer, Heidelberg (2009)
25. Kratsch, S., Wahlström, M.: Preprocessing of min ones problems: A dichotomy. In: Abramsky, S., Gavoille, C., Kirchner, C., Meyer auf der Heide, F., Spirakis, P.G. (eds.) ICALP 2010. LNCS, vol. 6198, pp. 653–665. Springer, Heidelberg (2010)

26. Kratsch, S., Wahlström, M.: Compression via matroids: a randomized polynomial kernel for odd cycle transversal. In: Rabani [29], pp. 94–103
27. Lokshtanov, D.: Wheel-free deletion is W[2]-hard. In: Grohe, M., Niedermeier, R. (eds.) IWPEC 2008. LNCS, vol. 5018, pp. 141–147. Springer, Heidelberg (2008)
28. Nemhauser, G.L., Trotter Jr., L.E.: Vertex packings: Structural properties and algorithms. Mathematical Programming 8(2), 232–248 (1975)
29. Rabani, Y. (ed.): Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012. SIAM, Kyoto (2012)
30. Thomassé, S.: A $4k^2$ kernel for feedback vertex set. ACM Transactions on Algorithms 6(2) (2010)
31. Yap, C.-K.: Some consequences of non-uniform conditions on uniform classes. Theor. Comput. Sci. 26, 287–300 (1983)