Machine Characterizations of the Classes of the W-hierarchy

Yijia Chen* Jörg Flum†

April 7, 2003

Abstract

We give machine characterisations of the parameterized complexity classes of the W-hierarchy. Moreover, for every class of this hierarchy, we present a parameterized halting problem complete for this class.

1. Introduction

Parameterized complexity theory provides a framework for a refined complexity analysis of algorithmic problems that are intractable in general. Central to the theory is the notion of *fixed-parameter tractability*, which relaxes the classical notion of tractability, polynomial time computability, by admitting algorithms whose runtime is exponential, but only in terms of some *parameter* that is usually expected to be small. As a complexity theoretic counterpart, a theory of *parameterized intractability* has been developed. In classical complexity theory, the notion of NP-completeness is central to a nice and simple theory for intractable problems. Unfortunately, the world of parameterized intractability is more complex: there is a big variety of seemingly different classes of parameterized intractability. Nevertheless, it can be argued that the classes $\mathrm{W}[1], \mathrm{W}[2], \ldots$ of the W-hierarchy together with some other classes like W[P] correspond to NP in classical complexity theory. In particular, all these classe are defined by parameterized variants of the NP-complete satisfiability problem. Unfortunately, the definition of these classes by means of complete problems makes it not easy to understand them. The authors of [4] tried to remedy this situation by presenting machine characterizations for some of the classes. But as they remarked "it remains an interesting open problem to find natural machine characterizations for the classes $\mathrm{W}[t]$ for $t \geq 2$". In this paper we obtain such characterizations. For this purpose, it is advantageous to use model-checking problems for fragments of first-order logic known to be complete for these classes.

By definition, a parameterized problem is fixed-parameter tractable, if it is decidable by a (deterministic) algorithm in at most $f(k) \cdot p(n)$ steps for some computable function f and some polynomial p. Here k denotes the size of the parameter and n the size of the input. Problems in any of the "intractable" parameterized complexity classes mentioned so far, also are decidable in at most $f(k) \cdot p(n)$ steps but by a *nondeterministic* algorithm. Thus, a first attempt to characterize one of these intractable classes by machines consists in considering nondeterministic algorithms that perform at most $f(k) \cdot p(n)$ steps but restricting further the number of nondeterministic steps. This approach led to the following results in [4]: A problem is in W[P] if and only if it decidable in at most $f(k) \cdot p(n)$ steps by an algorithm whose number of nondeterministic steps is bounded in terms of the parameter. Similarly, a problem is in $\mathrm{W}[1]$, if in addition the nondeterministic steps are performed at the end of the computation.

Already in [4], nondeterministic random access machines turned out to be the appropriate machine model in order to get clear formulations of the characterizations. In their nondeterministic steps these machines are able to guess natural numbers ($\leq f(k) \cdot p(n)$); these numbers are considered as names of objects. To obtain machine characterizations of the classes of the W-hierarchy we have to consider the corresponding alternating random access machines, but at the same time we have to ensure that the programs only have access to (the properties of) the elements named by the guessed numbers and not to the numbers themselves.

*Email: `chen@zermelo.mathematik.uni-freiburg.de`.
†Email: `Joerg.Flum@math.uni-freiburg.de`.

In [2], Cesati and Di Ianni prove that the halting problem for multitape nondeterministic Turing machines is W[2]-hard by reducing the parameterized dominating set problem to it: the machine first guesses the elements of a dominating set and then checks that they really constitute a dominating set. An analysis of the use of and the access to the guessed elements in this algorithm helped the authors to find the machine characterizations of the classes of the W-hierarchy. In Section 4 we present a (short) proof of membership in W[2] of the halting problem for multitape nondeterministic Turing machines by reducing it to a model-checking problem in W[2]. This result was obtained in [2] by different means.

As the corresponding proof in [4] shows, the machine characterization of W[1] is closely related to the W[1]-completeness of the halting problem for nondeterministic Turing machines, parameterized by the number of steps. For $t \geq 2$, the W[t]-complete halting problem we present in the last section refers to alternating Turing machines with oracles and it is not so closely related to the corresponding machine characterization but to a logical model-checking problem complete for W[t].

2. Preliminaries

In this section we recall some definitions and results and fix our notations.

2.1. Turing Machines. A nondeterministic Turing M machine is a tuple $(\Sigma, Q, q_0, q_{\text{acc}}, \delta)$. By default, M has no input tape and one work tape (bounded to the left and unbounded to the right). Σ is the *alphabet* of M, in addition M uses the *blank symbol* $*$. Q is the set of *states*, $q_0, q_{\text{acc}} \in Q$ are the *initial state* and the *accepting state*, respectively. Finally, δ is the *transition relation* consisting of tuples (q, a, q', a', h), written in the form $qa \to q'a'h$, where $q, q' \in Q$, $a, a' \in \Sigma \cup \{*\}$, and $h \in \{-1, 0, 1\}$ with the obvious meaning.

2.2. Relational Structures and First-order Logic. A *vocabulary* τ is a finite set of relation symbols. Each relation symbol has an *arity*. A *(relational) structure* $\mathcal{A}$ of vocabulary τ, or τ-structure, consists of a set A called the *universe*, and an interpretation $R^{\mathcal{A}} \subseteq A^r$ of each r-ary relation symbol $R \in \tau$. We synonymously write $\bar{a} \in R^{\mathcal{A}}$ or $R^{\mathcal{A}}\bar{a}$ to denote that the tuple $\bar{a} \in A^r$ belongs to the relation $R^{\mathcal{A}}$. For example, we view a *directed graph* as a structure $\mathcal{G} = (G, E^{\mathcal{G}})$, whose vocabulary consists of one binary relation symbol E. $\mathcal{G}$ is an (undirected) *graph*, if $E^{\mathcal{G}}$ is irreflexive and symmetric.

The class of all first-order formulas is denoted by FO. They are built up from atomic formulas using the usual boolean connectives and existential and universal quantification. Recall that *atomic formulas* are formulas of the form $x = y$ or $Rx_1 \dots x_r$, where $x, y, x_1, \dots, x_r$ are variables and R is an r-ary relation symbol. For $t \geq 1$, Σ_t denotes the class of all FO-formulas of the form

$$\exists x_{11} \dots \exists x_{1k_1} \forall x_{21} \dots \forall x_{2k_2} \ \dots \ Qx_{t1} \dots Qx_{tk_t} \ \theta,$$

where $Q = \forall$ if t is even and $Q = \exists$ otherwise, and where ψ is quantifier-free. Π_t-formulas are defined analogously starting with a block of universal quantifiers. Let $t, u \geq 1$. A formula φ is $\Sigma_{t,u}$, if it is Σ_t and all quantifier blocks after the leading existential block have length $\leq u$.

If Φ is a class of formulas, then $\Phi[\tau]$ denotes the class of all formulas of vocabulary τ in Φ. If $\mathcal{A}$ is a τ-structure, $a_1, \dots, a_n$ are elements of the universe A of $\mathcal{A}$, and $\varphi(x_1, \dots, x_n) \in \text{FO}[\tau]$ is a formula whose free variables are among $x_1, \dots, x_n$, then we write $\mathcal{A} \models \varphi(a_1, \dots, a_n)$ to denote that $\mathcal{A}$ satisfies φ if the variables $x_1, \dots, x_n$ are interpreted by $a_1, \dots, a_n$, respectively.

The proof of the following lemma, that will be used several times in this paper, is quite easy.

Lemma 1. *Let $\bar{x} = x_1 \dots x_r$ and $\bar{y} = y_1 \dots y_s$ be two sequences of variables that have no variable in common. Assume that $\varphi_1(\bar{x}), \dots, \varphi_m(\bar{x})$ and $\psi_1(\bar{x}, \bar{y}), \dots, \psi_m(\bar{x}, \bar{y})$ are formulas in* FO[τ] *and that $Q_1, \dots, Q_r, Q'_1, \dots, Q'_s \in \{\forall, \exists\}$. If $\mathcal{A}$ is a τ-structure such that $\mathcal{A} \models \forall \bar{x} \neg(\varphi_i \wedge \varphi_j)$ for $i \neq j$, then*

$$\mathcal{A} \models Q_1 x_1 \dots Q_r x_r \bigwedge_{i=1}^{m} (\varphi_i \to Q'_1 y_1 \dots Q'_s y_s \psi_i) \longleftrightarrow Q_1 x_1 \dots Q_r x_r Q'_1 y_1 \dots Q'_s y_s \bigwedge_{i=1}^{m} (\varphi_i \to \psi_i).$$

2.3. Parameterized Complexity. A *parameterized problem* is a set $Q \subseteq \Sigma^* \times \Pi^*$, where Σ and Π are finite alphabets. If $(x, y) \in \Sigma^* \times \Pi$ is an instance of a parameterized problem, we refer to x as the *input* and to y as the *parameter*. We usually denote the length of the input string x by n and the length of the paramter string y by k.

For example, for a class S of structures and a class L of first-order formulas, p-MC(S, L)

$$\text{p-MC(S, L)} \quad := \quad \{(\mathcal{A}, \varphi) \mid \mathcal{A} \in \text{S},\ \varphi \text{ a sentence in L, and } \mathcal{A} \models \varphi\}$$

is the *parameterized model-checking problem for* S *and* L; mostly, for easier readability, we present parameterized problems in the following form:

p-MC(S,L)	
Input:	$\mathcal{A} \in$ S.
Parameter:	φ sentence in L.
Problem:	$\mathcal{A} \models \varphi$?

Definition 2. A parameterized problem $Q \subseteq \Sigma^* \times \Pi^*$ is *fixed-parameter tractable*, if there is a computable function $f : \mathbb{N} \to \mathbb{N}$, a polynomial p, and an algorithm that, given a pair $(x, y) \in \Sigma^* \times \Pi^*$, decides if $(x, y) \in Q$ in at most $f(k) \cdot p(n)$ steps.

FPT denotes the complexity class consisting of all fixed-parameter tractable parameterized problems.

Complementing the notion of fixed-parameter tractability, there is a theory of parameterized intractability. It is based on the following notion of parameterized reduction:

Definition 3. An FPT-*reduction* from the parameterized problem $Q \subseteq \Sigma^* \times \Pi^*$ to the parameterized problem $Q' \subseteq (\Sigma')^* \times (\Pi')^*$ is a mapping $R : \Sigma^* \times \Pi^* \to (\Sigma')^* \times (\Pi')^*$ such that:

(1) For all $(x, y) \in \Sigma^* \times \Pi^*$: $(x, y) \in Q \iff R(x, y) \in Q'$.

(2) There exists a computable function $g : \mathbb{N} \to \mathbb{N}$ such that for all $(x, y) \in \Sigma^* \times \Pi^*$, say with $R(x, y) = (x', y')$, we have $k' \leq g(k)$ (where $k = |y|$ and $k' = |y'|$).

(3) There exist a computable function $f : \mathbb{N} \to \mathbb{N}$ and a polynomial p such that R is computable in time $f(k) \cdot p(n)$.

We write $Q \leq^{\text{FPT}} Q'$ if there is an FPT-reduction from Q to Q' and set

$$[Q]^{\text{FPT}} := \{Q' \mid Q' \leq^{\text{FPT}} Q\}.$$

For a class C of parameterized problems, we let

$$[\text{C}]^{\text{FPT}} := \bigcup_{Q \in \text{C}} [Q]^{\text{FPT}}.$$

Denote by GRAPH the class of all (finite) graphs and by STR the class of all finite structures. For the purposes of this paper, the most appropriate way to define the parameterized complexity classes of the W-hierarchy is the following (cf. [7] and [8]):

Definition 4. For $t \geq 1$, W$[t]$ is the class of all parameterized problems that, for some $u \geq 1$, are FPT-reducible to p-MC(GRAPH, $\Sigma_{t,u}$), that is,

$$\text{W}[t] = [\{\text{p-MC(GRAPH}, \Sigma_{t,u}) \mid u \geq 1\}]^{\text{FPT}}.$$

The following equivalent characterization of W$[t]$ is well-known:

Proposition 5. W$[t] = [\{$p-MC(STR, $\Sigma_{t,u}[\tau]) \mid \tau$ vocabulary$\}]^{\text{FPT}}$.

At various places of this paper we tacitly make use of the following remark:

Remark 6. Sometimes, in order to show that a given parameterized problem is in W$[t]$, we will present a reduction to p-MC(STR, $\Sigma_{t,u}[\tau]$) for some vocabulary τ also containing a fixed finite number of constant symbols or even, we will consider a reduction where the number of constants in the vocabulary depends on the parameter. This will allow to express properties in a more readable fashion. If we have a fixed finite number of constant symbols, they can be eliminated by using appropriate unary relations that are singletons; then, in the $\Sigma_{t,u}$-formula to be defined, the constants are replaced by variables that are existentially quantified (in the first block) and get their right value using the corresponding relations. If the number of constants symbols depends on the parameter, e.g., we use constants for $0, 1, \ldots, k$, in order to stay within a fixed vocabulary (independent of k) these constants can be eliminated by a singleton relation for 0 and the binary successor relation on $\{0, 1, \ldots, k\}$ and again by existentially quantified variables.

At some places we will refer to the parameterized complexity classes of A-hierarchy (cf. [8]):

Definition 7. For $t \geq 1$, A$[t]$ = [p-MC(GRAPH, Σ_t)]$^{\text{FPT}}$.

3. Machine Characterization

In [4], machine characterizations of the classes W[1], A$[t]$ for $t \geq 1$, and W[P] using nondeterministic and alternating random access machines (RAMs) were presented. The nondeterministic RAMs are based on the standard random access machines (cf. [9]). The model was non-standard when it came to nondeterminism. Instead of allowing the machines to nondeterministically choose one bit, or an instruction of the program to be executed next, the authors allowed them to nondeterministically choose a natural number, more precisely, there was an additional instruction "GUESS i j" whose semantics was: Guess a natural number less than or equal to the number stored in register i and store it in register j. In [4] it was remarked: "While this form of nondeterminism may seem unnatural at first sight, we would like to argue that it is very natural in many typical "applications" of nondeterminism. For example, a nondeterministic algorithm for finding a clique in a graph guesses a sequence of vertices of the graph and then verifies that these vertices indeed form a clique. Such an algorithm is much easier described on a machine that can guess the numbers representing the vertices of a graph at once, rather than guessing their bits."

The alternating RAMs, in addition to the instructions of the form "GUESS i j" (denoted by "EXISTS i j" in the context of alternating machines) also had "FORALL i j" instructions. The semantics was defined as usually for alternating machines. The computations of alternating machines suitable for A$[t]$ have a parameter-bounded final part which contains all the EXISTS- and FORALL-instructions and has at most $t-1$ alternations (see [4] for the precise statement). For W$[t]$ it seems not to suffice (see Remark 11 below and Section 6) to bound the length of the blocks without alternation, but we have to restrict the access to the numbers guessed in the EXISTS- and FORALL-instructions: as just remarked in the algorithm for finding a clique, we view these numbers as labels of certain objects and the type of machine we are going to introduce only has access to the properties of these objects and not directly to the labels.

We turn to the precise defintion of the random access machines we are going to use and that we call W-RAMs. A W-RAM has the

- the *standard registers* $0, 1, \ldots$, their contents are denoted by $r_0, r_1, \ldots$, respectively.
- the *guess registers* $0, 1, \ldots$, their contents are denoted by $g_0, g_1, \ldots$, respectively.

All registers have initial value 0. Sometimes we denote g_{r_i}, i.e., the contents of the guess register whose index is the content of the ith standard register, by $g(r_i)$

The W-RAM has all the standard instructions for the standard registers (cf. Section 2.6 of [9]; e.g., the arithmetic operations are addition, subtraction, and division by two (rounded off)). Moreover, it has four additional instructions:

Instruction	Semantics
EXISTS $\uparrow j$	guess a natural number $\leq r_0$ and store it in the r_jth guess register
FORALL $\uparrow j$	guess a natural number $\leq r_0$ and store it in the r_jth guess register
JGEQUAL $i\ j\ c$	if $g(r_i)$=$g(r_j)$, then jump to the instruction with label c
JGZERO $i\ j\ c$	if $r_{\langle g(r_i),g(r_j)\rangle} = 0$, then jump to the instruction with label c.

Here,

$$\langle\ ,\ \rangle : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$$

is any simple coding of ordered pairs of natural numbers by natural numbers such that $\langle i, j\rangle \leq (1 + \max\{i, j\})^2$ and $\langle 0, 0\rangle = 0$. Of course, the semantics of the EXISTS- and FORALL-instructions are the same, but we view them as existential and universal instructions, respectively. All other instructions are said to be deterministic. If the machine stops, it *accepts* its input, if $r_0 = 0$, otherwise it rejects it.

The following lemma, whose proof is immediate, is crucial for the main theorem of this section; it shows that the contents of the standard registers only depend on the sequence of executed instructions:

Lemma 8. *Assume that, for a given input, we have two (partial) computations on a W-RAM. If the same sequence of instructions is carried out in both computations, then the contents of the standard registers will be the same.*

Definition 9. A program $\mathbb{P}$ for a W-RAM is an *AW-program*, if there is a computable function f and a polynomial p such that for every input (x, k) with $|x| = n$ the program $\mathbb{P}$ on every run

(1) performs at most $f(k) \cdot p(n)$ steps;

(2) at most $f(k)$ steps are existential or universal;

(3) at most the first $f(k) \cdot p(n)$ standard registers are used;

(4) at every point of the computation the registers contain numbers $\leq f(k) \cdot p(n)$.

The promised machine characterization of W$[t]$ reads as follows:

Theorem 10. *Let Q be a parameterized problem and $t \geq 1$. Then Q is in* W$[t]$ *if and only if there is a $u \geq 1$ and there are a computable function h and an AW-program $\mathbb{P}$ for a W-RAM such that $\mathbb{P}$ decides Q and such that for every run of $\mathbb{P}$ on an instance (x, k) of Q as input*

– *all existential and universal steps are among the last $h(k)$ steps of the computation,*

– *there are at most $t - 1$ alternations between existential and universal states, and the first guess step is existential,*

– *every block without alternations, besides the first one, contains at most u guess steps.*

Proof: Assume first that $Q \in W[t]$, then for some $u \geq 1$,

$$Q \leq^{\text{FPT}} \text{p-MC(GRAPH}, \Sigma_{t,u}).$$

Hence there are computable functions f and g, a polynomial $p \in \mathbb{N}[x]$, and an algorithm $\mathbb{A}$ assigning to every (x, k), in time $\leq f(k) \cdot p(n)$, a graph $\mathcal{G} = \mathcal{G}_{x,k}$ and a sentence $\varphi = \varphi_{x,k} \in \Sigma_{t,u}$, say,

$$\varphi = \exists x_{11} \ldots \exists x_{1k_1} \forall x_{21} \ldots \forall x_{2k_2} \ldots Q x_{t1} \ldots Q x_{tk_t} \psi,$$

with $k_2, \ldots, k_t \leq u$, with $|\varphi| \leq g(k)$, and with a quantifier-free ψ, such that

$$Qxk \quad \iff \quad \mathcal{G} \models \varphi.$$

The claimed AW-program $\mathbb{P}$ for a W-RAM, on input (x, k), proceeds as follows:

1. It computes the graph $\mathcal{G} = (G, E^{\mathcal{G}})$, say with $G = \{1, \ldots, m\}$, and stores its adjacency matrix in the standard registers:

$$r_{\langle i,j \rangle} = 0 \quad \Longleftrightarrow \quad E^{\mathcal{G}} ij.$$

2. It computes φ.

3. It checks whether $\mathcal{G} \models \varphi$.

To carry out point 3, the program $\mathbb{P}$, using the EXISTS- and FORALL-instructions guesses the values of the quantified variables. Then, it checks the quantifier-free part using the JGEQUAL- and JGZERO-instructions. Clearly, the number of steps needed for point 3 can be bounded by $h(k)$ for some computable h. Hence, all existential and universal steps are among the last $h(k)$ steps of the computation.

Now assume that $\mathbb{P} = (\pi_1, \ldots, \pi_m)$ is an AW-program deciding Q and that $u \geq 1$, h, and $\mathbb{P}$ have the properties stated at the right side of the equivalence claimed in the theorem. Moreover, choose for the program $\mathbb{P}$ the function f and the polynomial p according to Definition 9. We claim that $Q \in \mathrm{W}[t]$. By Proposition 5, it suffices to show that $Q \leq^{\mathrm{FPT}}$ p-MC(STR, $\Sigma_{t,2\cdot u}[\tau]$) for some vocabulary τ. The set of instruction numbers of $\mathbb{P}$ is $\{1, \ldots, m\}$, more precisely, π_i is the instruction of $\mathbb{P}$ with instruction number i. We denote instruction numbers (= potential contents of the program counter) by $c, c_1, \ldots$ and finite sequences of instruction numbers by $\bar{c}$. $\ell(\bar{c})$ denotes the last instruction number of the sequence $\bar{c}$, and $\bar{c} \setminus \ell(\bar{c})$ the sequence obtained from $\bar{c}$ by omitting its last member. Let

$$C := \bigcup_{0 \leq r \leq h(k)-1} \{1, \ldots, m\}^r$$

and

$$N := \{0, 1, \ldots, f(k) \cdot p(n)\}.$$

Fix an instance (x, k) of Q. We look for a structure $\mathcal{A}$ and a $\Sigma_{t,2\cdot u}$-sentence φ such that

$$\mathbb{P} \text{ accepts } (x, k) \quad \Longleftrightarrow \quad \mathcal{A} \models \varphi.$$

Let $\bar{c}_0$ be the sequence of instruction numbers of the deterministic part of $\mathbb{P}$ on input (x, k) ending with the instruction number of the first existential instruction. As universe A of $\mathcal{A}$ we take $A := C \cup N$. Moreover, in $\mathcal{A}$ there are the binary relation $\leq^{\mathcal{A}}$, the natural ordering on N, and ternary relations $R^{\mathcal{A}}$ and $T^{\mathcal{A}}$ defined by

$$\begin{array}{lll} R^{\mathcal{A}}\bar{c}ij & \Longleftrightarrow & \bar{c} \in C,\ i, j \in N, \text{and if } \mathbb{P}, \text{ on input } (x,k), \text{ carries out the sequence} \\ & & \text{of instructions } \bar{c}_0\, \bar{c} \setminus \ell(\bar{c}_0\, \bar{c}), \text{ then } r_i = j. \\ T^{\mathcal{A}}\bar{c}ij & \Longleftrightarrow & \bar{c} \in C,\ i, j, \langle i,j \rangle \in N, \text{and if } \mathbb{P}, \text{ on input } (x,k), \text{ carries out the sequence} \\ & & \text{of instructions } \bar{c}_0\, \bar{c} \setminus \ell(\bar{c}_0\, \bar{c}), \text{ then } r_{\langle i,j \rangle} = 0. \end{array}$$

Moreover, we have a constant for 0. This finishes the definition of $\mathcal{A}$, which can be constructed within the time allowed by an FPT-reduction.

We turn to the definition of φ. First, we fix $\bar{c} \in C$: Let $i = i(\bar{c})$ be the number of blocks without alternations and $j = j(\bar{c})$ be the number of guesses in the ith block of the sequence of instructions determined by $\bar{c}_0\bar{c} \setminus \ell(\bar{c}_0\bar{c})$. If $i \leq t$ and if each block, besides the first one, has length $\leq u$, we introduce a formula

$$\varphi_{\bar{c}}(\bar{x}_1, \ldots, \bar{x}_{i-1}, x_{i,1} \ldots x_{i,j}) \tag{1}$$

where $\bar{x}_1 := x_{1,1}, \ldots, x_{1,h(k)}$ and $\bar{x}_s := x_{s,1} \ldots x_{s,u}$ for $s = 2, \ldots, i-1$ with the intuitive meaning

> if a partial run of $\mathbb{P}$ has $\bar{c}_0\, \bar{c}$ as sequence of instructions numbers and if every variable $x_{i',j'}$ displayed in (1) has, as value, the j'th guess of the i'th block (and $x_{i',j'} = 0$, if there was no such guess), then there is an accepting continuation of this run.

Clearly then, for $\varphi := \varphi_{\bar{c}_0}$, we have

$$\mathbb{P} \text{ accepts } (x,k) \quad \iff \quad \mathcal{A} \models \varphi.$$

For $\bar{c} \in C$ of maximal length, $|\bar{c}| = h(k) - 1$, we set (recall that, by definition, a computation accepts its input, if $r_0 = 0$ at the end of the computation)

$$\varphi_{\bar{c}} := \begin{cases} \text{TRUE}, & \text{if } \pi_{\ell(\bar{c})} = \text{STOP (i.e., } \pi_{\ell(\bar{c})} \text{ is the STOP-instruction) and } R\bar{c}\,0\,0 \\ \text{FALSE}, & \text{otherwise.} \end{cases}$$

If $\bar{c} \in C$ and $|\bar{c}| < h(k) - 1$, we assume that $\varphi_{\bar{c}'}$ has already been defined for all $\bar{c}'$ with $|\bar{c}| < |\bar{c}'|$. The definition depends on the type of the instruction of $\mathbb{P}$ with instruction number $\ell(\bar{c})$.

If $\pi_{\ell(\bar{c})} = \text{STOP}$, then again

$$\varphi_{\bar{c}} := \begin{cases} \text{TRUE}, & \text{if } R\bar{c}\,0\,0 \\ \text{FALSE}, & \text{otherwise.} \end{cases}$$

The definition of $\varphi_{\bar{c}}$ is simple for the standard instructions, e.g., if $\pi_{\ell(\bar{c})} = \text{STORE} \uparrow u$" (i.e., "$\pi_{\ell(\bar{c})} = r_{r_u} := r_0$"), then

$$\varphi_{\bar{c}} := \varphi_{\bar{c}\,\ell(\bar{c})+1}.$$

We give the definitions for the new instructions: If $\pi_{\ell(\bar{c})} = \text{EXISTS} \uparrow u$, then (recall that $i = i(\bar{c})$ and $j = j(\bar{c})$)

$$\varphi_{\bar{c}} := \begin{cases} \exists x_{i,j+1} \exists y (R\bar{c}\,0\,y \wedge x_{i,j+1} \leq y \wedge \varphi_{\bar{c}\,\ell(\bar{c})+1}), & \text{if } i \text{ is odd and } j < k \\ \exists x_{i+1,1} \exists y (R\bar{c}\,0\,y \wedge x_{i+1,1} \leq y \wedge \varphi_{\bar{c}\,\ell(\bar{c})+1}), & \text{if } i \text{ is even, } i < t, \text{ and } j = k \\ \exists x_{i+1,1} \exists y (R\bar{c}\,0\,y \wedge x_{i+1,1} \leq y \wedge \varphi_{\bar{c}\,\ell(\bar{c})+1} \wedge & \\ \qquad\qquad x_{i,j+1} = 0 \wedge \ldots \wedge x_{i,u} = 0), & \text{if } i \text{ is even, } i < t, \text{ and } j < k \\ \text{FALSE}, & \text{otherwise.} \end{cases}$$

The definition is similar for instructions of the type FORALL $\uparrow u$, but then the variables are quantified universally.

Assume $\pi_{\ell(\bar{c})} = \text{JGEQUAL } u\ v\ c$ ". We need $g(r_u)$ and $g(r_v)$. Determine the actual contents u_0 and v_0 of the uth and the vth standard register, i.e., u_0 and v_0 with $R^{\mathcal{A}}\bar{c}uu_0$ and $R^{\mathcal{A}}\bar{c}vv_0$. Consider the sequence of instructions given by $\bar{c}$ and determine the last instructions in it of the form FORALL $\uparrow w$ or EXISTS $\uparrow w$ such that at that time $r_w = u_0$, say, it is the j_0th guess in the i_0th block. Similarly, let the j_1th guess in the i_1th block be the last instruction of the form FORALL $\uparrow w$ or EXISTS $\uparrow w$ such that at that time $r_w = v_0$ (the case that such instructions do no exist are treated in the obvious way). Then set

$$\varphi_{\bar{c}} := (x_{i_0,j_0} = x_{i_1,j_1} \to \varphi_{\bar{c}\,c}) \wedge (\neg x_{i_0,j_0} = x_{i_1,j_1} \to \varphi_{\bar{c}\,\ell(\bar{c})+1}).$$

Assume $\pi_{\ell(\bar{c})} = \text{JGZERO } u\ v\ c$. As in the preceding case, let x_{i_0,j_0} and x_{i_1,j_1} denote the actual values of the r_uth and the r_vth guess register, respectively. Then set

$$\varphi_{\bar{c}} := (T\bar{c}x_{i_0,j_0}x_{i_1,j_1} \to \varphi_{\bar{c}\,c}) \wedge (\neg T\bar{c}x_{i_0,j_0}x_{i_1,j_1} \to \varphi_{\bar{c}\,\ell(\bar{c})+1}).$$

As already mentioned above, we set $\varphi := \varphi_{\bar{c}_0}$. By Lemma 1, one easily verifies that φ is equivalent to a $\Sigma_{t,2\cdot u}$-formula. Clearly, the size of φ can be bounded in terms of $h(k)$ and

$$\begin{aligned} Qxk \quad &\iff \quad \mathbb{P} \text{ accepts } (x,k) \\ &\iff \quad \mathcal{A} \models \varphi, \end{aligned}$$

which gives the desired reduction showing that $Q \in \mathrm{W}[t]$. □

Remark 11. If in the preceding theorem we use the alternating RAMs of [4], the contents of some standard registers will not only depend on the sequence $\bar{c}$ of instructions applied so far. Therefore, for some standard instructions one needs further quantifiers or function symbols to determine the current value of registers.

4. W$[2]$ and multitape machines

Among the many known W[1]-complete problems (cf. [6]), of course, the most generic one is the halting problem for nondeterministic Turing machines parameterized by the number of steps (cf. [1]):

p-HPN	
Input:	A nondeterministic Turing machine M.
Parameter:	$k \in \mathbb{N}$.
Problem:	Does M accept the empty word in at most k steps?

Surprisingly, the same problem for nondeterministic Turing machines with several tapes is W[2]-complete (cf. [3] and [2]). Mike Fellows pointed out this result to the second author and encouraged him to look for a machine characterization of W[2]. In this section we present a simple proof that the halting problem for multitape machines is in W[2]; it avoids the equality W[2] $=$ W*[2] (cf. [5]), used in [2] and for which no simple proof is known.

Proposition 12. *For some vocabulary τ,* p-HPNMT $\leq^{\text{FPT}}$ MC(STR, $\Sigma_{2,3}[\tau]$).

Here p-HPNMT denotes the halting problem for nondeterministic multitape Turing machines:

p-HPNMT	
Input:	A nondeterministic Turing machine M with an arbitrary finite number of tapes.
Parameter:	$k \in \mathbb{N}$.
Problem:	Does M accept the empty word in at most k steps?

Proof: Let M be a nondeterministic Turing machine with w_0 (work) tapes. We aim at a structure $\mathcal{A} = \mathcal{A}_{M,k}$ and a $\Sigma_{2,3}$-sentence $\varphi = \varphi_{M,k}$ such that

$$(M, k) \in \text{p-HPNMT} \quad \iff \quad \mathcal{A} \models \varphi.$$

The instructions of M have the form

$$q\,(a_1, \ldots, a_{w_0}) \;\to\; q'(a'_1, \ldots, a'_{w_0})\,(h_1, \ldots, h_{w_0})$$

where $a_1, \ldots, a_{w_0}, a'_1, \ldots, a'_{w_0} \in \Sigma \cup \{*\}$ and $h_1, \ldots, h_{w_0} \in \{-1, 0, 1\}$. Let T be the set of tuples $(b_1, \ldots, b_{w_0}) \in (\Sigma \cup \{*\})^{w_0}$ and H the set of tuples $(h_1, \ldots, h_{w_0}) \in \{-1, 0, 1\}^{w_0}$ occurring in instructions of M.

The structure $\mathcal{A}$ has the universe

$$A := Q \cup (\Sigma \cup \{*\}) \cup \{-1, 0, 1, \ldots, \max\{w_0, k\}\} \cup T \cup H.$$

We need the natural ordering relation $\leq^{\mathcal{A}}$ on $\{-1, 0, 1, \ldots, \max\{w_0, k\}\}$, the 5-ary relation $D^{\mathcal{A}}$ (the "transition relation") and the ternary relation $P^{\mathcal{A}}$ (the "projection relation") defined by

$$\begin{aligned} D^{\mathcal{A}} q t q' t' h \quad &\iff \quad qt \to q't'h \text{ is an instruction of } M \\ P^{\mathcal{A}} w b a \quad &\iff \quad 1 \leq w \leq w_0,\ b \in T \cup H,\ b = (b_1, \ldots, b_{w_0}), \text{ and } b_w = a. \end{aligned}$$

Moreover, we have constant symbols for the initial stated q_0, the accepting state q_{acc}, for $*$, and for $-1, w_0, 0, 1, \ldots, k$ (cf. Remark 6).

The formula φ we aim at, will express that there is an accepting run of length $\leq k$ (w.l.o.g. of length $= k$); among others, it will contain the variables $q_i, t_i, q'_i, t'_i, h_i$ for $i = 1, \ldots, k$, representing the ith instruction $q_i\, t_i \to q'_i\, t'_i\, h_i$ applied in the run; φ has the form

$$\exists q_1\, t_1\, q'_1\, t'_1\, h_1 \ldots \exists q_k\, t_k\, q'_k\, t'_k\, h_k\, (\varphi_{\text{init}}(q_1, t_1) \wedge \bigwedge_{i=1}^{k} D q_i\, t_i\, q'_i\, t'_i\, h_i \wedge \bigwedge_{i=1}^{k-1} q'_i = q_{i+1} \wedge q_k = q_{\text{acc}} \wedge \psi),$$

where $\varphi_{\text{init}}(q_1, t_1) := (q_1 = q_0 \wedge \forall w(1 \leq w \leq w_0 \rightarrow Pwt_1*))$ and where ψ is a universal formula expressing that the sequence of instructions can be applied: For this purpose, for $i = 1, \ldots, k$, we introduce quantifier-free formulas

$$\varphi_i^L(w, p, x, \bar{v}_i) \quad \text{and} \quad \varphi_i^S(w, p, \bar{v}_i)$$

with $\bar{v}_i := q_1, t_1, q'_1, t'_1, h_1, \ldots, q_{i-1}, t_{i-1}, q'_{i-1}, t'_{i-1}, h_{i-1}$ and with the meaning

> if, starting with the empty tape the sequence of instructions $\bar{v}_i$ has been carried out, then the pth cell of the wth tape contains the letter x,

and

> if, starting with the empty tape the sequence of instructions $\bar{v}_i$ has been carried out, then the head of the wth tape scans the pth cell,

respectively. Then, as ψ, we can take

$$\psi \quad := \quad \forall w \forall p \forall x \bigwedge_{i=1}^{k} \Big((\varphi_i^L(w, p, x, \bar{v}_i) \wedge \varphi_i^S(w, p, \bar{v}_i)) \rightarrow Pwt_i x \Big).$$

The simultaneous defintion of φ_i^L and φ_i^P by induction on i is routine:

$$\begin{aligned}
\varphi_1^L(w, p, x) \quad &:= \quad (1 \leq w \leq w_0 \wedge 1 \leq p \leq k \wedge x = *); \\
\varphi_1^S(w, p) \quad &:= \quad (1 \leq w \leq w_0 \wedge p = 1); \\
\varphi_{i+1}^L(w, p, x, \bar{v}_{i+1}) \quad &:= \quad (1 \leq w \leq w_0 \wedge 1 \leq p \leq k) \wedge \\
&\qquad ((\neg \varphi_i^S(w, p, \bar{v}_i) \wedge \varphi_i^L(w, p, x, \bar{v}_i)) \vee (\varphi_i^S(w, p, \bar{v}_i) \wedge Pwt'_i x)); \\
\varphi_{i+1}^S(w, p, \bar{v}_{i+1}) \quad &:= \quad (1 \leq w \leq w_0 \wedge 1 \leq p \leq k) \wedge \\
&\qquad (Pwh_i(-1) \rightarrow \bigvee_{l=2}^{k} (\varphi_i^S(w, l, \bar{v}_i) \wedge p = l - 1)) \wedge \\
&\qquad (Pwh_i 0 \rightarrow \varphi_i^S(w, p, \bar{v}_i)) \wedge \\
&\qquad (Pwh_i 1 \rightarrow \bigvee_{l=1}^{k-1} (\varphi_i^S(w, l, \bar{v}_i) \wedge p = l + 1)).
\end{aligned}$$

Using Lemma 1 one easily verifies that φ is (logically equivalent to) a $\Sigma_{2,3}$-sentence. □

Corollary 13. *The halting problem* p-HPNMT *for multitape machines is* W[2] *complete.*

Proof: By Proposition 5 and the preceding proposition, p-HPNMT is in W[2]. To show that p-HPNMT is W[2]-hard we recall the proof of [3] that the parameterized dominting set problem p-DS can be reduced to p-HPNMT. The essential problem is to find a corresponding machine in the time allowed by an FPT-reduction.

Suppose $(\mathcal{G}, k)$ is an instance of p-DS with $G = \{a_1, \ldots, a_n\}$. Let the multitape machine M have $n + 1$ tapes, numbered by 0 to n, and let $\Sigma := G \cup \{\text{yes}\}$ be its alphabet. In the first step all heads move one cell to the right. In the next $2 \cdot k$ steps, only the 0th head is active: it (nondeterministically) writes k elements of G on its tape, say $b_1, \ldots, b_k$ (the elements of the intended dominating set), and goes back to the cell containing b_1. In the next k moves the 0th head again reads these elements, in the corresponding jth step the ith head checks whether $a_i = b_j$ or Ea_ib_j; in the affirmative case the ith head prints "yes" and moves to the right, in the negative case it neither prints nor moves; finally, the machine moves all heads one cell to the left and accepts, if the heads on the tapes number 1 to n read "yes". Clearly,

$$(\mathcal{G}, k) \in \text{p-DS} \quad \iff \quad (M, 3 \cdot (k + 1)) \in \text{p-HPNMT}.$$

□

5. Complete halting problems for the W-hierarchy

As already mentioned at the beginning of Section 4, the halting problem for nondeterministic Turing machines, parameterized by the number of steps, is a quite generic W[1]-complete problem. The corresponding halting problems for alternating machines yield complete problems for the classes of the A-hierarchy. Indeed, for $t \geq 1$, we have $\mathrm{A}[t] = [\text{p-HPA}_t]^{\mathrm{FPT}}$ (cf. [8]), where

p-HPA$_t$	
Input:	An alternating Turing machine M whose initial state is existential.
Parameter:	$k \in \mathbb{N}$.
Problem:	Does M accept the empty word in at most k steps with at most t alternations?

One would guess that in order to obtain complete problems for the classes of the W-hierarchy one has to bound by a fixed constant the number of steps of all non-alternating blocks but the first one. More precisely, for $t, u \geq 1$, consider the following problem:

p-HPA$_{t,u}$	
Input:	An alternating Turing machine M whose initial state is existential.
Parameter:	$k \in \mathbb{N}$.
Problem:	Does M accept the empty word in at most k steps with at most t alternations, where every block of steps without alternation, besides the first one, has length $\leq u$?

Essentially along the lines of the proof of p-HPA$_t \in \mathrm{A}[t]$ in [8], one can show that p-HPA$_{t,u} \in \mathrm{W}[t]$. But the corresponding hardness proof does not seem to go through. Recall that in order to establish that $\text{p-MC}(\text{GRAPH}, \Sigma_t) \leq^{\mathrm{FPT}}$ p-HPA$_t$, given a graph and a Σ_t-sentence φ, one constructs an alternating Turing machine that first associates values to the quantifiers in φ (the values for existentially quantified variables are chosen in existential states, the values for universally quantified variables in universal states) and that then checks whether the selected variables satisfy the quantifier-free part of φ, the *quantifier-free check*. Of course, a $\Sigma_{t,u}$-prefix would yield an alternation sequence according to p-HPA$_{t,u}$, but the number of steps needed for the quantifier-free check cannot be bounded in terms of t and u as required in p-HPA$_{t,u}$, it depends on φ. Therefore, we add suitable oracles to the Turing machines that carry out the quantifier-free check in a single step. Of course, we have to add them in such a way that the corresponding halting problem still is in $\mathrm{W}[t]$.

As in an alternating machine, the set Q of states of an *alternating Turing machine M with oracle* is the disjoint union of the set of universal states Q_u and the set of existential states Q_e. The "oracle states" $q_?$, q_y, and q_n are all contained in Q_u or all in Q_e. Let Σ be the vocabulary of M and let $O \subseteq \Sigma^*$ be a language, the "oracle language". M^O, the machine M with oracle O, in state $q_?$ will check if the word to the left of the cell scanned by its head is in O and will change to the "yes state" q_y or to the "no state" q_n (without printing a letter nor moving its head).

For a graph $\mathcal{G} = (G, E^{\mathcal{G}})$ and a first-order formula $\psi(x_1, \dots, x_p)$, let $O(\mathcal{G}, \psi)$ be the following set of words over G, $O(\mathcal{G}, \psi) \subseteq G^*$:

$$O(\mathcal{G}, \psi) := \{a_1 \dots a_p \mid \mathcal{G} \models \psi(a_1, \dots, a_p)\}.$$

For $t, u \geq 1$, we consider the halting problem p-HPAO$_{t,u}$ for alternating Turing machines with oracle:

p-HPAO$_{t,u}$	
Input:	An alternating Turing machine M with oracle whose initial state is existential and a graph $\mathcal{G} = (G, E^{\mathcal{G}})$ such that G is a subset of the alphabet of M.
Parameter:	$k \in \mathbb{N}$ and a quantifier-free formula $\psi(x_1, \dots, x_p)$.
Problem:	Does $M^{O(\mathcal{G},\psi)}$ accept the empty word in at most k steps with at most t alternations, where every block of steps without alternation, besides the first one, has length $\leq u$?

Theorem 14. *For* $t \geq 1$,

$$\mathrm{W}[t] = [\{\text{p-HPAO}_{t,u} \mid u \geq 1\}]^{\mathrm{FPT}}.$$

Proof: For one direction, we show

$$\text{p-MC(GRAPH}, \Sigma_{t,u}) \quad \leq_{\mathrm{FPT}} \quad \text{p-HPAO}_{t,u+1}.$$

Suppose given a graph $\mathcal{G} = (G, E^{\mathcal{G}})$ and a $\Sigma_{t,u}$-formula φ,

$$\varphi \quad = \quad \exists x_{11} \dots \exists x_{1k_1} \forall x_{21} \dots \forall x_{2k_2} \dots Q x_{t1} \dots Q x_{tk_t} \psi(x_{11}, \dots, x_{tk_t})$$

with $k_2, \dots, k_t \leq u$ and with quantifier-free ψ. The desired reduction assigns to $\mathcal{G}$ and φ

$$\text{the number } k := k_1 + \ldots + k_{t-1} + (k_t + 1) \quad \text{and} \quad \text{the quantifier-free formula } \psi$$

as parameters, and, as inputs, the graph $\mathcal{G}$ and an alternating machine M with oracle: The alphabet of M contains the elements of G as letters. $M^{O(\mathcal{G},\psi)}$ first writes down elements $a_{11}, \dots, a_{1k_1}, \dots, a_{t1}, \dots, a_{tk_t}$ of G using existential and universal states alternatively; then, $M^{O(\mathcal{G},\psi)}$ asks its oracle $O(\mathcal{G}, \psi)$, if

$$\mathcal{G} \models \psi(a_{11}, \dots, a_{1k_1}, \dots, a_{t1}, \dots, a_{tk_t}).$$

The states $q_?$, q_y, and q_n are existential if t is odd, and universal if t is even; moreover, q_y is the accepting state of the machine. Clearly,

$$\mathcal{G} \models \varphi \quad \iff \quad ((M, \mathcal{G}), (k, \psi)) \in \text{p-HPAO}_{t,u+1},$$

which gives the desired reduction.

For the other direction we show that p-HPAO$_{t,u}$ $\leq^{\mathrm{FPT}}$ MC(STR, $\Sigma_{t,4\cdot u}[\tau]$) and thus, that p-HPAO$_{t,u} \in \mathrm{W}[t]$ by Proposition 5. For this purpose, let M, $\mathcal{G}$ (as inputs) and k, ψ (as parameters) be given as in the definition of p-HPAO$_{t,u}$. By adding suitable "dummy transitions" we obtain an alternating Turing machine M^+ with oracle such that

$$((M, \mathcal{G}), (k, \psi)) \in \text{p-HPAO}_{t,u} \quad \iff \quad \begin{array}{l} (M^+)^{O(\mathcal{G},\psi)} \text{ accepts the empty word in } k + (t-1) \cdot u \text{ steps} \\ \text{with } t \text{ alternations, the first block being existential and of} \\ \text{length } k\text{, and the remaining blocks of length } u\text{; moreover,} \\ (M^+)^{O(\mathcal{G},\psi)} \text{ performs at most } k \text{ nondummy steps.} \end{array}$$

For this purpose, besides the transitions of M, for every state $q \in Q$, $q \neq q_?$, and every $a \in \Sigma$, the machine M^+ has a dummy transition

$$qa \to qa0$$

(which allows to expand a block). And for every transition

$$qa \to q'a'm \tag{2}$$

with existential states q and q', M^+ has a new universal state $q_{(q,a)}$ and three new transitions, the first two ones being dummy and the last one nondummy,

$$\begin{array}{c} qa \to q_{(q,a)}a0 \\ q_{(q,a)}a \to q_{(q,a)}a0 \\ q_{(q,a)}a \to q'a'm \end{array}$$

(which allow to insert a universal block in an existential one). And similar states and transitions are added, if q and q' in (2) are universal. We denote by Q^+ and δ^+ the set of states and the set of transitions of M^+, respectively.

We look for a structure $\mathcal{A}$ and a $\Sigma_{t,4\cdot u}$-sentence φ such that

$$((M, \mathcal{G}), (k, \psi)) \in \text{p-HPAO}_{t,u} \quad \Longleftrightarrow \quad \mathcal{A} \models \varphi.$$

Let $A := G \cup Q^+ \cup (\Sigma \cup \{*\}) \cup \{-1, 0, 1, \dots, k\}$. The relations of $\mathcal{A}$ are a unary relation $G^{\mathcal{A}}$ (for the universe of the graph), the edge relation of $\mathcal{G}$, the succesor relation $S^{\mathcal{A}}$ on $\{-1, 0, 1, \dots, k\}$, relations $L^{\mathcal{A}}$, $R^{\mathcal{A}}$, $N^{\mathcal{A}}$, and $D^{\mathcal{A}}$ for the transitions with "left head movement", "right head movement", "nondummy transitions with no head movement" and dummy transitions; more precisely,

$$\begin{array}{rcl} L^{\mathcal{A}} & := & \{(q, a, q', a') \mid (q, a, q'a', -1) \in \delta^+\}; \\ R^{\mathcal{A}} & := & \{(q, a, q', a') \mid (q, a, q'a', 1) \in \delta^+\}; \\ N^{\mathcal{A}} & := & \{(q, a, q', a') \mid (q, a, q'a', 0) \text{ nondummy transition in } \delta^+\}; \\ D^{\mathcal{A}} & := & \{(q, a, q', a) \mid (q, a, q'a, 0) \text{ dummy transition in } \delta^+\} \end{array}$$

Moreover, we need constants for the numbers $-1, 0, 1, \dots, k$, for the blank symbol $*$, for the initial state q_0, for the accepting state q_{acc}, and for the oracle states $q_?, q_y$, and q_n (cf. Remark 6).

We turn to the definition of the formula φ. First, for ($i = 1$ and $0 \leq j \leq k$) or ($2 \leq i \leq t$ and $1 \leq j \leq u$) we define formulas

$$\varphi_{i,j}(q, h, y_1, \dots, y_k, s)$$

with the intuitive meaning

> if, after a partial run of $(M^+)^{O(\mathcal{G},\psi)}$, the jth step in the ith alternating block has been performed, $(M^+)^{O(\mathcal{G},\psi)}$ is in the state q, its head scans the hth cell, $y_1, \dots, y_k$ are the contents of the first k cells of the work tape, and s is the number of nondummy steps performed so far, then there is an accepting extension of the run satisfying the conditions required in the equivalence above.

We set

$$\varphi_{t,k}(q, h, y_1, \dots, y_k, s) \quad := \quad (q = q_{\text{acc}} \wedge 1 \leq h \leq k \wedge s \leq k).$$

Assume that $\varphi_{i',j'}$ has been defined for all i', j' with $i < i'$ or ($i = i'$ and $j < j'$). Then, $\varphi_{i,j}$ is defined by cases: For odd i (i.e., i denotes an existential block) and $j < k$, $\varphi_{i,j}(q, h, y_1, \dots, y_k, s)$ is the formula:

$$\begin{array}{l} \exists q' \exists h' \exists y' \exists s' \bigwedge_{l=1}^{k} \Big(h = l \to \big((Lqy_l q'y' \vee Rqy_l q'y' \vee Nqy_l q'y' \vee Dqy_l q'y_l \vee q = q_?) \wedge \\ \quad (Lqy_l q'y' \to (Sh'h \wedge Sss' \wedge \varphi_{i,j+1}(q', h', y_1, \dots, y_{l-1}, y', y_{l+1}, \dots, y_k, s'))) \wedge \\ \quad (Rqy_l q'y' \to (Shh' \wedge Sss' \wedge \varphi_{i,j+1}(q', h', y_1, \dots, y_{l-1}, y', y_{l+1}, \dots, y_k, s'))) \wedge \\ \quad (Nqy_l q'y' \to (Sss' \wedge \varphi_{i,j+1}(q', h, y_1, \dots, y_{l-1}, y', y_{l+1}, \dots, y_k, s'))) \wedge \\ \quad (Dqy_l q'y_l \to \varphi_{i,j+1}(q', h, y_1, \dots, y_k, s)) \wedge \\ \quad ((q = q_? \wedge (l \neq p \vee \bigwedge_{u=1}^{l} \neg Gy_u \vee \neg\psi(y_1, \dots, y_l))) \to (Sss' \wedge \varphi_{i,j+1}(q_n, h, y_1, \dots, y_k, s'))) \wedge \\ \quad ((q = q_? \wedge l = p \wedge \bigwedge_{u=1}^{l} Gy_u \wedge \psi(y_1, \dots, y_l)) \to (Sss' \wedge \varphi_{i,j+1}(q_y, h, y_1, \dots, y_k, s')))) \Big). \end{array}$$

For even i and $j = k$ we take the same formula, but replace $\varphi_{i,j+1}$ by $\varphi_{i+1,1}$ everywhere. We leave the remaining cases to the reader.

Finally, we take as φ the formula

$$\exists q \exists h \exists y_1 \ldots \exists y_k \exists s(q = q_0 \wedge h = 1 \wedge (y_1 = * \wedge \ldots \wedge y_k = *) \wedge s = 0 \wedge \varphi_{1,0}(q, h, y_1, \ldots, y_k, s).$$

By Lemma 1, one easily verifies that φ is (equivalent to) a $\Sigma_{t,4\cdot u}$-formula. $\square$

6. Conclusions

Most standard complexity classes like LOGSPACE, NLOGSPACE, PTIME, NPTIME, the classes of the polynomial hierarchy, or PSPACE have definitions in terms of machines. By contrast, originally nearly all parameterized complexity classes containing intractable problems were defined via complete problems. In [4], machine characterizations of W[1], W[P], and of A[t] for $t \geq 1$ were presented; in this paper we derive such characterizations for W[t] for $t \geq 2$.

As mentioned at the beginning of Section 3, in [4] alternating RAMs and AW-programs for them were introduced, which have "unrestricted access" to the guessed numbers. For $t \geq 1$, denote by L[t] the class that satisfies Theorem 10 for these type of machines, i.e., L[t] is the class of parameterized problems Q such that there is a computable function h and an AW-program $\mathbb{P}$ for an alternating RAM deciding Q such that for every run of $\mathbb{P}$ on an instance (x, k) of Q as input

- all existential and universal steps are among the last $h(k)$ steps of the computation,
- there are at most $t - 1$ alternations between existential and universal states, and the first guess step is existential,
- every block without alternations, besides the first one, contains at most u guess steps.

Clearly, by [4] and Theorem 10, we have

$$\mathrm{W}[t] \subseteq \mathrm{L}[t] \subseteq \mathrm{A}[t].$$

We do not know, if W[t] $=$ L[t] or if L[t] $=$ A[t]. Along the lines of the proof of Theorem 15 in [4], one can show that L[t] $= [\{\text{p-HPA}'_{t,u} \mid u \geq 1\}]^{\text{FPT}}$, where p-HPA$'_{t,u}$ is the following halting problem for alternating Turing machines with existential, universal, and *deterministic* states (compare it with the halting problem p-HPA$_{t,u}$).

p-HPA$'_{t,u}$

Input:	An alternating Turing machine M with existential, universal, and deterministic states, whose initial state is existential.
Parameter:	$k \in \mathbb{N}$.
Problem:	Does M accept the empty word in at most k steps with at most t alternations between existential and universal steps, where every block of steps without alternation, besides the first one, contains at most u existential or at most u universal steps?

References

[1] L. Cai, J. Chen, R.G. Downey, and M.R. Fellows. On the parameterized complexity of short computation and factorization. *Archive for Mathematical Logic*, 36:321–337, 1997.

[2] M. Cesati. The Turing way to parameterized intractability, 2001. Submitted for publication.

[3] M. Cesati and M. Di Ianni. Computation models for parameterized complexity. *Math. Log. Quart.*, 43:179–202, 1997.

[4] Y. Chen, J. Flum, and M. Grohe. Bounded nondeterminism and alternation in parameterized complexity theory. To appear in *Proc. of the Conference on Computational Complexity*, 2003. Available at `http://www.dcs.ed.ac.uk/home/grohe/pub.html`.

[5] R.G. Downey and M.R. Fellows. Threshold dominating set and an improved characterization of W[2]. *Theoretical Computer Science*, 209:123–140, 1996.

[6] R.G. Downey and M.R. Fellows. *Parameterized Complexity*. Springer-Verlag, 1999.

[7] R.G. Downey, M.R. Fellows, and K. Regan. Descriptive complexity and the W-hierarchy. In P. Beame and S. Buss, editors, *Proof Complexity and Feasible Arithmetic*, volume 39 of *AMS-DIMACS Volume Series*, pages 119–134. AMS, 1998.

[8] J. Flum and M. Grohe. Fixed-parameter tractability, definability, and model checking. *SIAM Journal on Computing*, 31(1):113–145, 2001.

[9] C.H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.