Warsaw University
Faculty of Philosophy and Sociology

Leszek Aleksander Kołodziejczyk

Truth Definitions and Higher Order Logics in Finite Models

Ph.D. Thesis

Thesis Advisor:
Marcin Mostowski
Institute of Philosophy
Warsaw University

Warsaw
February 2005

Abstract

The first part of the thesis concerns the expressive power of higher order logics in finite models. We introduce the notion of *basic arity*, a higher order analogue of the *arity* of a second order relation, and use it to define some fragments of higher order prenex classes which turn out to have natural complexity-theoretical counterparts. We also show that for $1 \leqslant r \leqslant d$, the logic $Q_r(\Sigma^d_m)$, defined as the closure of the prenex class Σ^d_m under boolean operations and r-th order quantification, captures the complexity class

$$(\mathsf{r}-2)\mathsf{EXPSPACE}^{(\mathsf{d}-\mathsf{r})\Sigma_m\mathsf{EXP}}.$$

This extends the result that $Q_1(\Sigma^d_m)$ captures $\mathsf{LOGSPACE}^{(\mathsf{d}-1)\Sigma_m\mathsf{EXP}}$, which follows from a theorem of G. Gottlob.

In the second part of the thesis, we develop the theory of *finite model truth definitions*, introduced by M. Mostowski in an attempt to transfer the classical model-theoretic notion of *truth definition* to finite model theory. We prove that under quite weak assumptions, the existence of a finite model truth definition for a logic $\mathcal{L}$ in another logic $\mathcal{L}'$ depends exclusively on the expressive power of $\mathcal{L}$ and $\mathcal{L}'$, and not on their syntax. We use this to show that for a number of typical logics, there exist finite model truth definitions whose evaluation requires resources barely above the complexity classes captured by those logics.

Our general results about truth definitions can be applied in computational complexity theory. In particular, combined with a type of normal form result known form papers of I. Stewart and G. Gottlob, they yield new information about bounded query fragments of P^{NP}. Under a suitable notion of universal language (called *weak* in a recent paper on universal languages and diagonalization by Nash *et al.*), we obtain the following:

Let $m \geqslant 2$. Then:

(A) If there is a Δ^p_m universal language for NP, then for every r, there also is a Δ^p_muniversal language for $\mathsf{P}^{\mathsf{NP}[n^r]}$;

(B) If there are both a Σ^p_m and a Π^p_m universal language for NP, then for every r, there are both a Σ^p_m and a Π^p_m universal language for $\mathsf{P}^{\mathsf{NP}[n^r]}$.

(C) If there exists a superpolynomial time-constructible function f such that $\mathsf{NTIME}(f) \subseteq \Sigma^p_m \cap \Pi^p_m$, then there is a $\Sigma^p_m \cap \Pi^p_m$ universal language for $\mathsf{P}^{\mathsf{NP}[n^r]}$ (and hence $\mathsf{P}^{\mathsf{NP}[n^r]} \subsetneq \Sigma^p_m \cap \Pi^p_m$).

Each of these three results can be viewed as a strengthening of S. Mocas' theorem that for any r, $\mathsf{P}^{\mathsf{NP}[n^r]} \subsetneq \mathsf{NEXP}$.

Streszczenie

Pierwsza część pracy dotyczy siły wyrazu logik wyższych rzędów w modelach skończonych. Wprowadzamy pojęcie *arności bazowej*, stanowiące odpowiednik *arności* dla relacji wyższych rzędów, i definiujemy za jego pomocą pewne fragmenty klas preneksowych logik wyższych rzędów. Fragmenty te, jak się okazuje, mają naturalne odpowiedniki w teorii złożoności obliczeniowej. Pokazujemy też, że dla $1 \leqslant r \leqslant d$, logika $Q_r(\Sigma_m^d)$, czyli domknięcie klasy preneksowej Σ_m^d na operacje boolowskie i kwantyfikację r-tego rzędu, chwyta klasę złożoności

$$(\mathsf{r}-2)\mathsf{EXPSPACE}^{(\mathsf{d}-\mathsf{r})\Sigma_m\mathsf{EXP}}.$$

Uogólnia to wynik, iż $Q_1(\Sigma_m^d)$ chwyta $\mathsf{LOGSPACE}^{(\mathsf{d}-1)\Sigma_m\mathsf{EXP}}$, który jest wnioskiem z twierdzenia G. Gottloba.

W drugiej części pracy rozwijamy teorię *skończenie-modelowych definicji prawdy*, wprowadzonych przez M. Mostowskiego jako wariant klasycznego pojęcia *definicji prawdy* dostosowany do potrzeb teorii modeli skończonych. Dowodzimy, że przy stosunkowo słabych założeniach, istnienie skończenie-modelowej definicji prawdy dla logiki $\mathcal{L}$ w innej logice $\mathcal{L}'$ zależy wyłącznie od siły wyrazu $\mathcal{L}$ i $\mathcal{L}'$, a nie od ich składni. Korzystając z tego, pokazujemy, iż dla wielu typowych logik istnieją skończenie-modelowe definicje prawdy, których ewaluacja wymaga zasobów tylko nieznacznie wykraczających poza klasy złożoności chwytane przez te logiki.

Nasze ogólne rezultaty dotyczące definicji prawdy mają pewne zastosowania w teorii złożoności obliczeniowej. W szczególności, powiązane z pewnymi wynikami o postaciach normalnych (w stylu znanym z prac I. Stewarta i G. Gottloba), dostarczają nowej wiedzy o podklasach P^{NP} definiowanych przez ograniczenie liczby pytań do wyroczni. Przy stosownym pojęciu języka uniwersalnego (określanym mianem *słabego języka uniwersalnego* w niedawnej pracy Nasha i in. o językach uniwersalnych i diagonalizacji), otrzymujemy:

Niech $m \geqslant 2$. Wtedy:

(A) Jeśli Δ_m^p zawiera język uniwersalny dla NP, to zawiera też, dla dowolnego r, język uniwersalny dla $\mathsf{P}^{\mathsf{NP}[n^r]}$;

(B) Jeśli zarówno Σ_m^p, jak i Π_m^p zawiera język uniwersalny dla NP, to dla dowolnego r, zarówno Σ_m^p, jak i Π_m^p zawiera język uniwersalny dla $\mathsf{P}^{\mathsf{NP}[n^r]}$.

(C) Jeśli istnieje ponadwielomianowa czasowo konstruowalna funkcja f t. że $\mathsf{NTIME}(f) \subseteq \Sigma_m^p \cap \Pi_m^p$, to $\Sigma_m^p \cap \Pi_m^p$ zawiera język uniwersalny dla $\mathsf{P}^{\mathsf{NP}[n^r]}$ (a zatem $\mathsf{P}^{\mathsf{NP}[n^r]} \subsetneq \Sigma_m^p \cap \Pi_m^p$).

Każdy z tych trzech wyników można traktować jako wzmocnienie uzyskanego przez S. Mocas twierdzenia, iż dla dowolnego r, $\mathsf{P}^{\mathsf{NP}[n^r]} \subsetneq \mathsf{NEXP}$.

Acknowledgments

I would like to express my gratitude to all the numerous people who have helped me during my PhD studies. In particular, I thank my advisor, Marcin Mostowski, for introducing me to logic in general and to finite model theory, for his guidance during my work on this thesis, and for his ability to bear with such an exceptionally stubborn student; Zofia Adamowicz, for teaching me lots of things which did not have a direct influence on the results presented here, but certainly shaped my way of thinking about logic; and Konrad Zdanowski, for all the discussions over the last few years, and for reading and commenting on a preliminary version of the thesis.

I am very grateful to Paweł Konieczny for what would normally be called "technical assistance". However, as the circumstances in which I had to finish my work were, for various reasons, strongly *ab*normal, Paweł's help went well beyond that.

Finally, I thank my wife Magda and my parents for all their love and support.

Contents

Introduction

The present thesis concerns two very old logical themes put in a relatively new context.

Higher order logics, which grew out of Russell's type theory ([Rus08]), were originally considered a central part of logic. Later on, however, they became increasingly overshadowed by first order logic, which had a number of attractive metalogical properties they lacked, such as axiomatizability, compactness, or the Skolem-Löwenheim property. Another factor contributing to this downward trend was the growing perception of higher order logic as "set theory in disguise" ([Qui70]). Eventually, higher order logic acquired the reputation of being an esoteric subject. Most of whatever little attention it received was devoted to a variant (due to Henkin [Hen50]) which had a non-standard semantics and could essentially be viewed as a many-sorted first order theory. In the last few decades, interest in logics of higher order has again begun to grow — as evidenced, e.g., by the two survey articles [Lei94] and [BD01]. Nevertheless, their theory, especially in the case of third and higher order, still remains relatively underdeveloped.

The investigation of *truth definitions*, as a means of comparing the expressive power of various logics, was initialized in Tarski's seminal paper [Tar33]. Already in that paper, Tarski proved his celebrated theorem on the undefinability of truth: no (reasonable) logic closed under negation defines arithmetical truth for itself. Incidentally, another major result contained in [Tar33] concerned truth definitions for higher order logics: Tarski showed that for every n, $(n+1)$-st order logic defines truth for n-th order logic; hence, it is strictly more expressive. With time, the method of truth definitions became *the* diagonal technique for comparing the expressivity of logics — as an illustration, let us just recall the folklore result that each of the prenex classes of first and higher order logic, Σ^d_m, defines truth for itself, so it is not closed under negation and is strictly smaller than Σ^d_{m+1}. Even today, new

separations and hierarchicity results are being proved using truth definitions — see e.g. [HS00].

Our aim in the thesis is to study both higher order logics and truth definitions in the context of *finite model theory*. This branch of logic originated with Trakhtenbrot's theorem that the first order theory of finite models is not axiomatizable ([Tra50]). Unfortunately, due perhaps to the prejudices of the time, this negative result seemed to hold back further work on finite models rather than stimulate it. There were some results about finite structures in the 1960s and early 70s, but consistent development of finite model theory began with Fagin's theorem ([Fag74]), which established a connection between the logical definability of classes of finite models and their computational complexity. This development accelerated rapidly in the 1980s, with new results on the logic-complexity borderline coming from Immerman, Vardi, and numerous others. Today, after two decades of growth, finite model theory is an established research field. In the last few years, this field seems to be going through a crisis, with no clear research programme to unite it and no new groundbreaking results. However, connections with other fields are opening up — for example, [Ats02], [CK03], or [Kl04] link finite model theory with propositional proof complexity and bounded arithmetic. Thus, there is hope that the crisis will prove temporary.

Much of the work on finite models has dealt with fragments of second order logic — but the subject of higher orders has barely been touched. Similarly, some diagonal arguments have turned up in finite model theory — but this has not led to the general study of such arguments. So, one could reasonably say that in the setting of finite model theory, the topics of higher order logics and truth definitions are both well-explored, from one point of view, and seriously underresearched, from another.

In our thesis, we try to take a closer look at the underresearched regions within these topics. Our contribution is threefold. We prove two new expressibility results for fragments of finite order logic in finite models. We investigate a concept of finite model truth definitions, introduced by M. Mostowski, and obtain some rather general results on their behaviour. Finally, we use those results to prove some theorems in pure computational complexity theory.

A more comprehensive overview of our work follows.

Overview of results

The thesis comprises this introduction, four chapters and two appendices. Chapter 1 is preliminary and contains some basic information about the background and notation. Our results are presented in chapters 2, 3, and 4 — treating, respectively, higher order logics, truth definitions, and some applications of the latter to computational complexity theory. The contents of these chapters are described in more detail below. The appendices contain two constructions we decided to leave out from the main body of the thesis. Appendix A proves a normal form lemma needed in chapter 2. Appendix B gives an example of a finite model truth definition.

Higher order logics. If "higher" order is taken to mean "third or higher", then not much work has been done on higher order logics in finite models. [Ben62] may be thought of as the prehistory of such research. Later contributions include [Lei87] and, recently, [HT03]. Among other things, these papers characterized the expressive power of the prenex classes of higher order logics (the Σ^d_m classes). This characterization, which is difficult to attribute exactly[1], essentially remains the only well-known result about the power of higher order logics in finite models. Prior to our work, the one other result in that direction known to us followed from Gottlob's [Got97]. Corollary 4.9 and theorem 5.3 in that paper, taken together, characterize the power of the first order closure of a higher order prenex class.

In chapter 2 of our thesis, we prove two new theorems about the expressive power of higher order logics. One of them appeared in [Koł04b]. It refines the results about the power of prenex classes by introducing the notion of *basic arity* (an analogue of the second order notion of arity) and subdividing the Σ^d_m classes according to the basic arity of high order variables. The other result, which has not yet been published, generalizes some parts of [Got97] to describe the strength of a higher order prenex class closed under quantification of *any* fixed lower order. As corollaries of the two theorems, we obtain new logical characterizations of some well-known complexity classes, for example NE and $\mathsf{PSPACE}^{\mathsf{NP}}$.

The theorems are proved using standard tools: reasoning in the style of Fagin's theorem, census techniques as in [Hem89], and a type of normal form known from the work of Stewart ([Ste93]) and Gottlob ([Got97]).

[1] See the historical remark after proposition 2.2.

Truth definitions. Although diagonal arguments have been used in finite model theory for some time — [MP96] or [DH95] may serve as examples — the introduction of the very notion of "truth definition" into the context of finite model theory took place only recently, and is the independent work of two people. Raatikainen ([Raa00]) concentrates on the representability of truth-in-a-finite-model in first order arithmetical theories which have finite models. We shall not pursue that approach here. M. Mostowski, on the other hand ([Mos01], [Mos03]), transfers the original concept of truth definition into a more usual finite model setting. He proposes to deal with models which have a suitable amount of arithmetical structure (so that gödelization can be carried out), and lets a finite model truth definition be a formula which decides the truth of any sentence of a given logic in all sufficiently large models of this kind.

Inspired by the classical results of [Tar33], Mostowski proved some theorems on the existence of finite model truth definitions for various fragments of higher order logic. Our chapter 3 contains the (rather simple) proofs of some new, quite general results on finite model truth definitions. We prove that under some very mild conditions, the problem whether one logic has a finite model truth definition for another does not depend on the syntax of these logics, but only on their expressive power. This allows us to characterize the existence of finite model truth definitions in, and for, various logics in terms of computational complexity. We also offer some remarks on how the idea of finite model truth definitions relates to Vardi's ([Var82]) notion of *combined complexity*.

Essentially all the material of chapter 3 appeared in [Koł04b].

Applications. The final, and most technical, chapter of the thesis is devoted to applying our results about truth definitions to computational complexity theory. We deal with *bounded query* fragments of the polynomial hierarchy, especially with the $\mathsf{P}^{\mathsf{NP}[n^r]}$ classes (deterministic polynomial time with at most n^r queries to an NP oracle).

An intriguing feature of the $\mathsf{P}^{\mathsf{NP}[n^r]}$ classes is that they are the largest fragments of the polynomial hierarchy which have been separated from NEXP. Although it is widely conjectured that $\mathsf{PH} \subsetneq \mathsf{NEXP}$, not even $\mathsf{P}^{\mathsf{NP}} \subsetneq \mathsf{NEXP}$ has actually been proven. Moreover, the problem whether P^{NP} equals NEXP is known to have contradictory relativizations (see [BT94] or [BFFT01] for an oracle relative to which $\mathsf{P}^{\mathsf{NP}} = \mathsf{NEXP}$). On the other hand, Fu *et al.* [FLZ92]

managed to show $\mathsf{P}^{\mathsf{NP}[o(n)]} \subsetneq \mathsf{NEXP}$, and some time later, Mocas ([Moc96]) improved this to $\mathsf{P}^{\mathsf{NP}[n^r]} \subsetneq \mathsf{NEXP}$. The proofs of these theorems relativize.

Using a notion of universal language for a complexity class (known from [NIR03] as *weak* universal language), we prove three results of the form: if NP is "much smaller" than a certain fragment of PH (i.e. the fragment contains a universal language for NP), then so is $\mathsf{P}^{\mathsf{NP}[n^r]}$. Each of these results implies $\mathsf{P}^{\mathsf{NP}[n^r]} \subsetneq \mathsf{NEXP}$, so they may be understood as evidence that "the reason why" we are able to separate the $\mathsf{P}^{\mathsf{NP}[n^r]}$ classes, unlike P^{NP} itself, from NEXP lies not in their special relation to nondeterministic exponential time, but rather in their position within the polynomial hierarchy. We also discuss analogous results for other complexity classes in place of $\mathsf{P}^{\mathsf{NP}[n^r]}$.

Throughout chapter 4, we make extensive use of the "finite model truth definitions" framework developed in chapter 3. Other logical tools are also involved — in particular, as in chapter 2, we again need a normal form in the Stewart-Gottlob style. Still, some of our proofs could be transferred relatively easily to a standard complexity-theoretical setting, with just Turing machines, simulations etc. (in fact, we do put one of our arguments in such a form). Others, however — especially the work of section 4.3 — seem to use the logical machinery essentially. Eliminating the logic, even if possible, would make those arguments quite awkward and artificial.

Most, though not all, of the results in this chapter have been included in [Koł04a]. The major omission in that paper is part (A) of theorem 4.6.

Chapter 1

Preliminaries

This chapter contains the logical and complexity-theoretical preliminaries to the thesis. We explain the theoretical background needed for understanding our results, and fix some notation along the way.

The preliminaries are divided into two sections, the first of which is purely logical, while the second recalls some computation theory and complexity theory, and links these with logic by discussing the so-called descriptive complexity theory.

1.1 The logical framework

Models and vocabularies. We work over finite models with built-in arithmetic. That is, we assume two things. Firstly, the universe of a given model $\mathbf{M}$ is always the initial segment $M = \{0, \ldots, M-1\}$ of the natural numbers (note that we are using the standard set-theoretical representation of natural numbers, so e.g. $card(M) = M$). No infinite models are considered unless explicitly stated otherwise. No finite models with universes other than $\{0, \ldots, n-1\}$ are ever considered.

Secondly, the vocabulary always contains a fixed arithmetical subvocabulary σ_0 (consisting, say, of the predicates $+, \times, \leqslant$ and constants $0, MAX$), and the symbols of σ_0 are always interpreted in the standard way. So, over $\mathbf{M}$, MAX is to be interpreted as $M-1$, 0 is to be 0, and the predicate symbols are to represent restrictions of the appropriate relations from the standard model of arithmetic — e.g., $+(a, b, c)$ is to hold if and only if $a + b = c$.

All vocabularies are finite and relational, possibly with individual con-

stants (i.e., we disallow only function symbols). The usual letter for denoting vocabularies is σ, often accompanied by an index. We refer to models of vocabulary σ as *σ-models*. Thus, for example, models of the purely arithmetical vocabulary are σ_0-models.

The letters for predicates are P, R, S, etc., sometimes with indices and/or numerical superscripts to indicate the arity — thus, R^r is an r-ary predicate. The letter for constants is c. The way to denote the interpretation of a predicate R in $\mathbf{M}$ is "officially" $R^{\mathbf{M}}$, but we almost always drop the $\mathbf{M}$, leaving it intact only to avoid notational havoc (so, in most cases we use the same notation for predicates and for the relations interpreting them). Similarly for constants.

The vocabulary which results from expanding σ by, e.g., a relational symbol R will be denoted by $(\sigma + R)$. The result of expanding a model $\mathbf{M}$ by a relation R is $(\mathbf{M}, R)$. Similarly for constants.

Logics. For technical reasons, it will be convenient to adopt a very broad definition of "logic". Thus, a *logic* is any function $\mathcal{L}$ which assigns to a vocabulary σ a decidable set of words over some finite alphabet (the set of *$\mathcal{L}$-sentences over σ*) and a decidable relation $\models^{\mathcal{L}}_{\sigma}$ between σ-models and $\mathcal{L}$-sentences over σ (the *truth relation*, normally denoted by just $\models$ if no confusion arises). If φ is an $\mathcal{L}$-sentence over σ, then $MOD_\sigma(\varphi)$ (or simply $MOD(\varphi)$ if no danger of confusion arises) denotes the set of σ-models $\mathbf{M}$ such that $\mathbf{M} \models \varphi$. We say that $MOD_\sigma(\varphi)$ is *defined* by φ.

The syntax of first order logic (FO) and second order logic (SO) is well-known, as are the prenex classes of second order logic (Σ^1_m, Π^1_m). For the definitions, see e.g. any monograph on finite model theory, such as [EF95] or [Imm99] (not all general logic textbooks are suitable, since some of them do not discuss second order logic). We use lower-case letters from the end of the alphabet, such as x, y, z, possibly with indices, for first order variables. The notation for second order variables is the same as for predicates from a vocabulary — capital letters P, R, S, etc., sometimes with indices or numerical superscripts. In general, this does not lead to confusion.

Properties of logics. In many cases, we will need our logics to satisfy some closure conditions. These are defined in the natural way. For example, a logic $\mathcal{L}$ is *closed under negation* if for any $\mathcal{L}$-sentence φ over a vocabulary σ there is an $\mathcal{L}$-sentence ψ such that $MOD(\psi)$ is the set of those σ-models

which are not in $MOD(\varphi)$. As a slightly more subtle example, $\mathcal{L}$ is *closed under existential quantification* if the following holds: for any $\mathcal{L}$-sentence φ over $(\sigma + c)$ — c being an individual constant — there is an $\mathcal{L}$-sentence ψ over σ such that $MOD(\psi)$ consists of those $\mathbf{M}$ for which there is $n \in M$ with $(\mathbf{M}, n) \in MOD(\varphi)$.

If $\mathcal{L}$ and $\mathcal{L}'$ are logics, then $\mathcal{L} \leqslant \mathcal{L}'$ means that $\mathcal{L}'$ is at least as expressive as $\mathcal{L}$, i.e., all classes of models definable by an $\mathcal{L}$-sentence are also definable by an $\mathcal{L}'$-sentence. $\mathcal{L} \equiv \mathcal{L}'$ holds if $\mathcal{L} \leqslant \mathcal{L}'$ and $\mathcal{L}' \leqslant \mathcal{L}$. $\mathcal{L} < \mathcal{L}'$ holds if $\mathcal{L} \leqslant \mathcal{L}'$ but not $\mathcal{L} \equiv \mathcal{L}'$. For any vocabulary σ, $\mathcal{L} \leqslant_\sigma \mathcal{L}'$ means that $\mathcal{L}'$ is at least as expressive as $\mathcal{L}$ over vocabulary σ. $\mathcal{L} \equiv_\sigma \mathcal{L}'$ and $\mathcal{L} <_\sigma \mathcal{L}'$ are defined accordingly.

Finite variants. Let K and K' be any two sets — in particular, classes of models. We say that K' is a *finite variant of* K if the symmetric difference of K and K' is finite.

1.1.1 Representing concepts in finite models

In much of our work, the ability to represent arithmetical relations — typically infinite — in finite models will be essential. Clearly, no infinite relation can be *represented* in a single finite model in the usual sense of the term, so we follow [Mos01] and use a somewhat altered, "asymptotic" notion of representation:

Definition 1.1. A relation $R \subseteq \omega^n$ is *FM-represented* by the *FO*-formula $\varphi(x_1, \ldots, x_n)$ (of vocabulary σ_0) if and only if: for any $a_1, \ldots, a_n \in \omega$, $\varphi(a_1, \ldots, a_n)$ is true in almost all σ_0-models if $R(a_1, \ldots, a_n)$ holds, and false in almost all σ_0-models if $R(a_1, \ldots, a_n)$ does not hold. R is *FM-representable* if there is a formula which FM-represents it.

The following *FM-representability theorem* characterizes the FM-representable relations:

Theorem 1.2 ([Mos01]). *A relation $R \subseteq \omega^n$ is FM-representable if and only if it is of degree $\leqslant \mathbf{0}'$ (equivalently: recursive with an* RE *oracle, Δ^0_2 in the arithmetical hierarchy).*

One consequence of theorem 1.2 is that, just as in the standard model of arithmetic or in arithmetical theories, also in finite models we may freely

talk about operations and relations connected to the syntax of logics. Let us make this a bit more precise.

We choose a finite alphabet A so that we may treat sentences of any logic, and other syntactical objects such as variables, connectives etc. used in the more usual logics, as words over A. $A = \{0, 1\}$ is probably the most obvious choice. Recall that given any other alphabet B, we have a canonical translation f_B of words over B into words over $\{0, 1\}$ with just a linear blow-up in length: if $B = \{b_1, \ldots, b_m\}$, let $f_B(b_i)$ equal the lexicographically i-th $\log_m$-bit long[1] binary string, and extend this uniquely to a homomorphism $B^* \to \{0, 1\}^*$. We have $\mathrm{lh}(f_B(w)) = \log m \cdot \mathrm{lh}(w)$ (where $\mathrm{lh}(\cdot)$ stands for the length of a string).

We then fix a correspondence between words $w \in A^*$ and natural numbers $\ulcorner w \urcorner$ (their Gödel numbers). To be specific: for a given $w = w^1 \ldots w^k$ with $w^i \in \{0, 1\}$, we let $\ulcorner w \urcorner$ be the number whose binary representation has length $k + 1$ and is of the form $1w^1 \ldots w^k$. So, for example, $\ulcorner 00101 \urcorner$ would be 37.[2]

Via this correspondence, we are often going to treat the elements of our models (i.e. natural numbers) as elements of A^*. In this way, syntactical relations, such as "the first order formula x is the result of preceding formula y with an existential quantifier", become relations between numbers. Note that syntactical relations are decidable — therefore, we can "talk about them" simply by applying theorem 1.2 and using the formulae which FM-represent them.

Care must be taken, however, since FM-representation works "asymptotically": a formula φ which FM-represents some relation will only tell us whether a given tuple $\mathbf{a}$ is in the relation if we look at the truth value of $\varphi(\mathbf{a})$ in a sufficiently large model.

A refinement of the concept of FM-representability will appear in section 4.3.

[1] We use $\log n$ to denote $\lceil \log_2 n \rceil$, where log_2 is the usual (real-valued) binary logarithm function. So, for example, the binary representation of a number $n \neq 0$ (without leading zeroes) has exactly $\log(n + 1)$ bits.

[2] Another possible way of identifying natural numbers with words in A^* would be to refer to the results of [Zda05] (presented also in [KZ05]). It is shown there that the n-element initial segment of the natural numbers, with the usual arithmetical relations, and the n-element initial segment of A^*, with concatenation as the only relation and the characters from A named as constants, are mutually interdefinable, uniformly in n (in the case of A^*, initial segments are taken w.r.t. the standard length-lexicographic ordering).

1.2 Computability, complexity, and descriptive complexity

Throughout the thesis, we freely use the general concept of a Turing machine and the elementary notions of computability theory, such as those of a *recursive* (or *decidable*) relation, of a *recursively enumerable* (RE) relation. For background in computability theory, see e.g. [Sch93].

Essentially, we also freely refer to the basic notions and methods of complexity theory: the best-known computational resources and complexity classes, the notions of a (polynomial time many-one) reduction and complete problem, padding, simple clocking techniques etc. Nevertheless, we do review some of those basics briefly, mainly to fix the terminology and notation. Missing details may be found in [Pap94], [BDG95]. We also expand somewhat on the complexity theory we need. A vast majority of the complexity-theoretical topics we touch upon can be found in the two books mentioned above, or in [BDG90] (which is part II of [BDG95]).

General framework. We use multi-tape Turing machines with a separate read-only input tape.

Recall that a configuration of a Turing machine T can be viewed as a tuple

(state of T, positions of the heads on the input tape and worktapes, contents of the worktapes).

A computation of T is thus a sequence or tree of configurations, depending on whether T is deterministic or not.

We think of complexity classes as sets of *languages* over $\{0,1\}^*$, that is, as subsets of $\mathcal{P}(\{0,1\}^*)$. We do not need a precise definition of what makes a set of languages a complexity class; in fact, we may assume that a complexity class is just *any* subset of $\mathcal{P}(\{0,1\}^*)$. However, our main interest lies in the natural complexity classes defined using bounds on computational resources, as described below.

Basic resources and classes; constructibility. Let f be a function on the natural numbers. $\mathsf{DTIME}(f)$ is the class of problems (i.e. languages) which can be recognized by a deterministic Turing machine in time bounded by f (w.r.t. the length of the input string). $\mathsf{NTIME}(f)$ is defined analogously using

nondeterministic machines[3]. $\mathsf{DSPACE}(f)$ is the class of languages which can be recognized by a deterministic machine using at most f worktape cells.

A function f is *time-constructible* if there is a deterministic machine which halts in exactly $f(n)$ steps on any input of length n. Typically, it is only the the time-constructible functions which are used in the definitions of DTIME/NTIME classes, as only they are well-behaved as time bounds. There is an additional convention that time bounds f are assumed to satisfy $f(n) \geqslant n$ (so that the machine always has time to read the whole input). This convention is by no means universally accepted, but we *do* use it in the thesis.

There is a similar notion of constructibility for space: f is *space-constructible* if there is a machine which on any input of length n halts with exactly $f(n)$ worktape cells non-blank, and does not use any additional workspace during its computation.

"Normal" functions which grow at least as fast as the identity function, such as n^2, $n \cdot \log n$, $n!$, n^n etc., are both time- and space-constructible. Functions such as $\log n$ or $\log^2 n$ are space-constructible.

The most important and well-known complexity classes include the following:

- $\mathsf{P} = \bigcup_{k\in\omega} \mathsf{DTIME}(n^k)$
- $\mathsf{NP} = \bigcup_{k\in\omega} \mathsf{NTIME}(n^k)$
- $\mathsf{L} = \bigcup_{k\in\omega} \mathsf{DPSPACE}(k \cdot \log n)$
- $\mathsf{PSPACE} = \bigcup_{k\in\omega} \mathsf{DSPACE}(n^k)$.

The class L is also (perhaps more often) called LOGSPACE. The class P is frequently identified with the class of computationally feasible problems (this identification should be taken with a grain of salt, but it has nevertheless proven extremely useful and led to a lot of deep research). The question whether $\mathsf{P} = \mathsf{NP}$ is considered to be the most fundamental problem in theoretical computer science, and one of the most fundamental in mathematics in general.

For any class C, co-C consists of the complements of the languages in C.

[3]We sidestep the subtle issue of how exactly to define the time taken by a nondeterministic machine, since we only need the concept for time-constructible functions f (see below), and in their case all reasonable definitions of $\mathsf{NTIME}(f)$ will turn out to be equivalent anyway.

Oracle machines and relativized classes. We will often consider complexity classes defined by bounds on oracle computations, so we need to equip some of our machines with an oracle access mechanism. We use the *Ladner-Lynch* model of oracle computation (see [LL76]). This means that an oracle machine has a separate, *unique* write-only oracle tape for writing down the oracle queries, and the oracle tape is erased immediately after a query is made. Furthermore, in the case of space-bounded computation, the oracle tape is *not* subject to the space bound. So, for example, it may happen that a $\mathsf{DSPACE}(f)$ oracle machine asks 2^f-bit long queries.

A configuration of an oracle machine T should contain information about the oracle tape, so it is a tuple

(state of T, positions of the heads on the input tape and worktapes, contents of the worktapes, content of the oracle tape).

A *genuine configuration* is one where the oracle tape is empty.

If C is a complexity class defined using bounds on resources and L is a language, then C^L (C *relativized to* L) is the class of languages recognized by oracle machines which obey the bounds defining C and use L as their oracle. For example, P^L is the class of languages recognizable in deterministic polynomial time with oracle L. The class C^{D} is defined as the union of C^L for $L \in \mathsf{D}$. Of course, such notation should be used with caution, as it may happen that for two distinctly defined classes C_1 and C_2, $\mathsf{C}_1 = \mathsf{C}_2$ holds, but ${\mathsf{C}_1}^{\mathsf{D}} = {\mathsf{C}_2}^{\mathsf{D}}$ does not, or vice versa.

Alternating machines. Apart from the usual deterministic and nondeterministic Turing machines, we use a simple special case of alternating machines (see [CKS81]), the so-called Σ_m and Π_m machines, for $m \geqslant 1$. An alternating machine is like a nondeterministic machine, except that it has two kinds of non-final states, existential and universal, which influences the notion of acceptance. Acceptance is defined by induction on the computation tree, from the leaves upwards. A leaf, i.e. a final configuration, accepts iff the state is accepting. A configuration with an existential state accepts iff at least one of its sons in the computation tree accepts, and a configuration with a universal state accepts iff all of its sons do. The machine accepts a given input iff its initial configuration on that input accepts.

A Σ_m *machine* is an alternating machine which starts in an existential state and switches between existential and universal states at most $m - 1$

times on a single computation path (observe that a Σ_1 machine is essentially just a nondeterministic machine). A Π_m *machine* is defined dually, i.e., it starts in a universal state.

The polynomial hierarchy. PH, or the *polynomial hierarchy*, a very well-known hierarchy of classes above NP, is traditionally defined using relativizations: Σ_1^p is NP, Σ_{m+1}^p is $\mathsf{NP}^{\Sigma_m^p}$, and $\Pi_m^p = \mathsf{co}\text{-}\Sigma_m^p$ (the hierarchy itself is the union of all these levels). However, PH can also be defined using alternating machines: Σ_m^p is $\bigcup_{k\in\omega} \Sigma_m \mathsf{TIME}(n^k)$ (polynomial time on a Σ_m machine). Π_m^p can be defined as previously or as $\bigcup_{k\in\omega} \Pi_m \mathsf{TIME}(n^k)$. In our opinion, this definition transfers more elegantly to "higher order" analogues of the polynomial hierarchy (see chapter 2), and is more suited to descriptive complexity contexts (see the following subsection and chapter 2), so we prefer it to the oracle-based one.

Unfortunately, there is no natural definition of the intermediate levels of PH using alternating machines, so here we use the traditional definition: Δ_{m+1}^p is $\mathsf{P}^{\Sigma_m^p}$.

Exponential and other classes. To complete our minicatalogue of complexity classes, we note that there are two standard notions of nondeterministic exponential time: NE is as $\bigcup_{k\in\omega} \mathsf{NTIME}(2^{kn})$, NEXP is $\bigcup_{k\in\omega} \mathsf{NTIME}(2^{n^k})$. Of course, these classes also have their deterministic versions E and EXP, respectively, and space versions ESPACE and $\mathsf{EXPSPACE}$, respectively.

Some further complexity classes will be defined in the body of the thesis: superexponential classes and hierarchies will appear in chapter 2, whereas some bounded query fragments of PH will play a central role in chapter 4.

Reductions and complete problems. A very important notion in complexity theory is that of a polynomial-time reduction. A problem L is *polynomial-time (many-one) reducible* to L' if there is a polynomial-time computable function f on words such that $w \in L$ iff $f(w) \in L'$ (f is the *polynomial-time reduction* of L to L').

A problem L is *complete* for a class C if $L \in \mathsf{C}$ and any language in C is polynomial-time reducible to L. The theory of complete problems initiated with the seminal result that SAT, propositional formula satisfiability, is complete for NP ([Coo71]).

Of course, one may define many other notions of reducibility and completeness by modifying the restrictions on f.

Closure conditions; pairs and tuples. There are many natural closure conditions a complexity class may (or may not) satisfy. Two prominent examples are closure under complementation (the central problem here being whether $\mathsf{NP} = \mathsf{co\text{-}NP}$) and closure under various kinds of reductions (E and NE may serve as examples of superpolynomial complexity classes not closed under polynomial-time reductions).

One other closure condition we need to discuss is closure under conjunctions. Loosely speaking, C is *closed under conjunctions* if, for any language $L \in \mathsf{C}$, the problem: "given a list of strings, are they all in L?" is also in C.

To treat this more formally, we need to devise a way of coding ordered pairs or tuples of words over $\{0,1\}$ as single words over the same alphabet. If $w_1, \ldots, w_k$ are words, and each w_i is $w_i^1 \ldots w_i^{j_i}$, the w_i^j's being bits, we let $\langle w_1, \ldots, w_k \rangle$ be

$$w_1^1 w_1^1 \ldots w_1^{j_1} w_1^{j_1} 01 \ldots w_k^1 w_k^1 \ldots w_k^{j_k} w_k^{j_k} 01$$

(we double all bits in the w_i's and add 01 strings, treated as new # symbols, as separators). If we define $L\#$ as $\{w^1 w^1 \ldots w^j w^j 01 : w^1 \ldots w^j \in L\}$, then $(L\#)^*$ consists exactly of the tuples whose elements are all in L. So, we say that C is closed under conjunctions if $(L\#)^* \in \mathsf{C}$ for any $L \in \mathsf{C}$.

Most standard complexity classes, including all those defined above, are closed under conjunctions.

Hierarchy theorems and linear-speed up. We need to mention one further topic in complexity theory: the *hierarchy theorems*. Let us start by recalling some pieces of notation used for comparing the growth rates of functions:

Definition 1.3. Let f, g be functions (on the natural numbers).

- $f = O(g)$ if there exists a constant $C > 0$ such that $f(n) \leqslant C \cdot g(n)$ for almost all n;
- $f = o(g)$ if for any $c > 0$, $f(n) \leqslant c \cdot g(n)$ for almost all n;
- $f = \Theta(g)$ iff $f = O(g)$ and $g = O(f)$.

The hierarchy theorems are results which say roughly the following: under suitable conditions, a suitably larger bound on a given type of computational resource defines a strictly larger complexity class (i.e., strictly more languages can be recognized using resources bounded by the larger bound). The hierarchy theorem for DSPACE, originally proved by Hartmanis and Stearns, is arguably the most elegant:

Theorem 1.4 (see e.g. [BDG95]). *Let f and g be space-constructible functions which satisfy* $\log \leqslant f = o(g)$. *Then* $\mathsf{DSPACE}(f) \subsetneq \mathsf{DSPACE}(g)$.

There are also variants of this theorem for DTIME (here we need to replace the condition $f = o(g)$ by $f \cdot \log f = o(g)$), NTIME, and Σ_mTIME for any fixed m (most of these variants may be found in [BDG95]). An important feature of the hierarchy theorems is that they relativize. That is, given a language L, we could replace DSPACE in theorem 1.4 by DSPACE^L (deterministic space with oracle L) and the theorem would still hold. Similarly for other types of resources.

Often, we will loosely invoke "the hierarchy theorems", without giving further details, to justify that some complexity class is strictly smaller than another.

A result which, in a sense, goes in the opposite direction to hierarchy theorems is the *linear speed-up theorem*, also originally due to Hartmanis and Stearns: for any f such that the identity function is $o(f)$, it holds that $\mathsf{DTIME}(O(f))) \subseteq \mathsf{DTIME}(f)$; analogously for NTIME and Σ_mTIME. There also exists a similar result for space classes.

1.2.1 Descriptive complexity theory

Descriptive complexity theory (DCT, for short) is a field relating logic, in particular finite model theory, to computational complexity. Essentially all of the material in our thesis is either directly a part of, or connected to DCT. This subsection very briefly explains the basics of DCT. For more, refer to [Imm99]. Additional information may also be found in [EF95] or [Lib04].

The main idea of DCT is the following. Finite models are essentially combinatorial objects, so a class of finite models (of fixed vocabulary) can be identified with a computational problem. Viewed from such an angle, there are two natural notions of complexity associated with a class of models: one is its *descriptive complexity*, i.e. how hard it is to describe the class using logic, the other is the *computational complexity* of the class seen as computational

problem. The question to ask is: how are these complexities related? DCT is an attempt at an answer.

Moving on to the technicalities: given a fixed vocabulary σ, we want to code σ-models as words over $\{0,1\}$. The code for $\mathbf{M}$ is a word of length $O(M^k)$, where k is the maximal arity of a predicate in $\sigma \setminus \sigma_0$ ($k = 1$ if there is no such predicate). The code starts with $M - 1$ 1's and a 0, followed by codes for the interpretations of the non-arithmetical σ-symbols (in some fixed order). The code for $R^{\mathbf{M}}$, with R k-ary, is a M^k-bit string whose i-th bit is 1 iff the i-th tuple in M^k (ordered lexicographically) is in $R^{\mathbf{M}}$. The code for $c^{\mathbf{M}}$ is the $(\log M)$-bit long binary representation of $c^{\mathbf{M}}$ (with the appropriate number of leading zeroes).

Example. If σ is $(\sigma_0 + R^2 + c)$ and $\mathbf{M}$ is $(3, R^{\mathbf{M}}, 1)$ with $R^{\mathbf{M}}$ consisting just of $(0,1)$ and $(2,0)$, then the code for $\mathbf{M}$ is 11001000010001.

Once we identify σ-models with their codes, a class of σ-models becomes a language over $\{0,1\}^*$, so its computational complexity is well-defined. For technical reasons, however, we work under the following proviso: in calculating the complexity of a class of models, one considers the size of $\mathbf{M}$ to be M, and *not* the length of the code for $\mathbf{M}$, which, as noted, is in general polynomially larger. This is important in the case of complexity classes such as NE and the bounded query classes defined in chapter 4.

Now we come the the central definition of DCT. If $\mathcal{L}$ is a logic and C is a complexity class, we say that $\mathcal{L}$ *captures* C if:

for any σ and any class of σ-models $\mathcal{K}$, it holds that $\mathcal{K}$ is in C if and only if $\mathcal{K}$ is definable by an $\mathcal{L}$-sentence.

If every $\mathcal{L}$-definable class $\mathcal{K}$ is in $\mathcal{C}$, we say that *model checking for* $\mathcal{L}$ *is in* C. If every class in C is $\mathcal{L}$-definable, $\mathcal{L}$ is said to *capture at least* C.[4]

[4]It is more usual to define "$\mathcal{L}$ captures $\mathcal{C}$" and the other related notions over models with just a built-in linear ordering, and not built-in arithmetic. In our framework, some logics (such as first order logic) may capture larger complexity classes than in the usual one. However, if a logic contains deterministic transitive closure logic (DTC; cf. [Imm99]), it is able to define the arithmetical relations from the ordering, so its expressive power is the same in both frameworks. Most of the specific logics we consider (with the important exception of FO) do contain DTC.

The notion of capturing has also been studied in a context where *no* built-in relations are allowed (see chapter 12 of [Imm99] for more information). This has led to some of the most interesting research in DCT, but lies outside the scope of this thesis.

Naturally, all three definitions can be restricted to a particular vocabulary, resulting in "$\mathcal{L}$ *captures* C *over* σ" and so on.

DCT is essentially the study of which logics capture which complexity classes — also, under what circumstances (see e.g. the footnote after the definition of capturing), and what this implies for complexity theory proper.

The seminal result of DCT is Fagin's theorem:

Theorem 1.5 (([Fag74])). Σ^1_1 *captures* NP.

This was later extended by Stockmeyer:

Theorem 1.6 ([Sto77]). *For any* m, Σ^1_m *captures* Σ^p_m.

Further connections between logics and complexity classes were discovered later on. To list some of the canonical examples: *deterministic transitive closure logic* DTC captures L ([Imm87]), *least fixed point logic* LFP captures P ([Imm86],[Var82]), and *partial fixed point logic* PFP captures PSPACE ([Var82]). Other examples, both known and new, will be given in chapters 2 and 4.

Chapter 2

Higher Order Logics

We move on to one of the two main topics of the thesis: the theory of higher order logics in finite models.

We begin by reviewing the syntax and semantics of (a relational variant of) finite order logic, along with the basic results on its expressive power. In particular, we discuss the expressive power of the prenex classes of higher order logics, the so-called Σ^d_m classes.

Then, we introduce the notion of *basic arity*, a generalization of the concept of *arity* to higher orders, and define some fragments of the Σ^d_m classes by limiting the basic arity of the variables allowed. It turns out that a straightforward modification of the proof of Fagin's theorem allows us to characterize the expressive power of these fragments, denoted $[\Sigma^d_m]^{\leqslant k}$, in terms of computational complexity classes. The characterization yields, e.g., a natural logic capturing the complexity class NE.

The last section of the chapter is devoted to studying the logics which arise when a prenex fragment of a higher order logic is closed under quantification of some lower order. It is known that if we close a prenex fragment just under first order quantification, the resulting logic captures a relativized logarithmic space class. We generalize that theorem to the situation where closure under quantification of second, third, ... order is involved. To obtain the generalization, we need to prove an appropriately modified version of an important normal form lemma known from the literature.

2.1 Higher order logics: syntax, semantics, expressive power

The name "higher order logics" is normally used to refer to a number of fragments of finite order logic $\mathcal{L}^\omega$. Below, we define the syntax and semantics of a relational variant of finite order logic, and discuss its expressive power. For a much more comprehensive survey on higher order logics, refer to [Lei94]. Some additional information may also be found in [BD01].

The syntax and semantics of $\mathcal{L}^\omega$ are defined in analogy to those of second order logic, the crucial difference being that quantification over relations of arbitrary (finite) order is allowed.

The set **Typ** of *types* is defined inductively as the smallest set which satisfies: $\iota \in \mathbf{Typ}$, and for any $\tau_1, \ldots, \tau_k \in \mathbf{Typ}$, also $(\tau_1 \ldots \tau_k) \in \mathbf{Typ}$. The intended interpretation is that objects of type ι are individuals, i.e. elements of the universe of a given model, whereas objects of type $(\tau_1 \ldots \tau_k)$ are k-ary relations whose i-th argument is an object of type τ_i. The set of types is naturally stratified into orders, namely, $\mathrm{order}(\iota) = 1$ and $\mathrm{order}((\tau_1 \ldots \tau_k)) = max(\{\mathrm{order}(\tau_1), \ldots, \mathrm{order}(\tau_k)\}) + 1$.

The language of $\mathcal{L}^\omega$ has countably many variables for each $\tau \in \mathbf{Typ}$. The order of a variable is the order of its type. It is customary to use superscripts to indicate the type of a variable: $\mathbf{R}^\tau$ is a variable of type τ. We treat this convention rather loosely, feeling free to drop the superscripts whenever we find them unnecessary or bothersome. Throughout the chapter, we adhere to the general conventions concerning notation for first and second order variables (see the *Preliminaries*). Variables of third and higher (or unspecified) order are written in boldface.

The set of $\mathcal{L}^\omega$-formulae is defined just as for second order logic except that higher order variables are allowed. In the following definition, note that under our conventions regarding vocabularies, a *term* is either a first order variable or an individual constant from the vocabulary:

Definition 2.1. The set of $\mathcal{L}^\omega$*-formulae over vocabulary* σ is the smallest set X containing:

- $t_1 = t_2$, for any two terms t_1, t_2 (over σ);
- $P(t_1, \ldots, t_k)$, for any predicate P^k from σ and any terms $t_1, \ldots, t_k$;
- $R(t_1, \ldots, t_k)$, for any second order variable R^k and any terms $t_1, \ldots, t_k$;

- $\mathbf{S}(\mathbf{R}_1, \ldots, \mathbf{R}_k)$, for any variables $\mathbf{S}^{(\tau_1 \ldots \tau_k)}, \mathbf{R}_1^{\tau_1}, \ldots, \mathbf{R}_k^{\tau_k}$ of the appropriate types;
- $\neg\varphi, (\varphi \,\&\, \psi), (\varphi \vee \psi), (\varphi \Rightarrow \psi), (\varphi \equiv \psi)$, for any $\varphi, \psi \in X$;[1]
- $\forall\mathbf{S}\varphi$, $\exists\mathbf{S}\varphi$, for any $\varphi \in X$ and any variable $\mathbf{S}^\tau$ of any type τ.

The *atomic formulae* are the ones introduced by one of the first four clauses. The $\mathcal{L}^\omega$*-sentences* are just the $\mathcal{L}^\omega$-formulae without any free variables, where the standard notion of free variable is extended to higher orders in the obvious way.

Also the semantics for $\mathcal{L}^\omega$ is defined in the obvious way, via valuations and the satisfaction relation. Naturally, valuations are required to assign objects of appropriate type to the variables. So, given $\mathbf{M}$, we define the universe of type τ over $\mathbf{M}$ — denoted $\mathbf{M}_\tau$ — by induction on τ. The definition is unsurprising: $\mathbf{M}_\iota = M$, $\mathbf{M}_{(\tau_1 \ldots \tau_k)} = \mathcal{P}(\mathbf{M}_{\tau_1} \times \ldots \times \mathbf{M}_{\tau_k})$. A valuation in $\mathbf{M}$ is then any function val from the set of all variables to $\bigcup_{\tau \in \mathbf{Typ}} \mathbf{M}_\tau$ satisfying $val(\mathbf{R}^\tau) \in \mathbf{M}_\tau$ for any variable $\mathbf{R}^\tau$ of any type τ. The definition of satisfaction and truth is completely standard. For example, $\mathbf{M} \models \mathbf{S}(\mathbf{R}_1, \ldots, \mathbf{R}_k)[val]$ iff $(val(\mathbf{R}_1), \ldots, val(\mathbf{R}_k)) \in val(\mathbf{S})$.

The restriction of $\mathcal{L}^\omega$ to formulae whose variables are all of order $\leqslant d$ is called d-th order logic and denoted $\mathcal{L}^d$. It may be easily verified that $\mathcal{L}^1$ is simply FO, and $\mathcal{L}^2$ is simply SO.

An $\mathcal{L}^\omega$-formula is in *prenex normal form* if all quantifiers are at the beginning of the formula, forming a quantifier prefix which precedes the quantifier-free part (matrix) of the formula. We additionally assume that quantifiers of the highest order precede all the remaining quantifiers in the prefix. Every $\mathcal{L}^\omega$-sentence is equivalent to an $\mathcal{L}^\omega$-sentence in prenex normal form. The stronger statement, that every $\mathcal{L}^d$-sentence is equivalent to an $\mathcal{L}^d$-sentence in prenex normal form, is also true.

As recalled in the *Preliminaries*, it is well-known that second order formulae in prenex normal form comprise hierarchies whose levels are denoted Σ^1_m and Π^1_m. The definition of these hierarchies can be extended to higher order logics. The class Σ^d_m consists of those $\mathcal{L}^{d+1}$-formulae in prenex normal form in which the $(d+1)$-st order quantifiers are arranged into m alternating universal and existential blocks, starting with an existential block. Π^d_m is defined dually. It is clear from the previous paragraph that every $\mathcal{L}^{d+1}$-sentence is equivalent to a Σ^d_m sentence for some m.

[1] We feel free to skip the parentheses whenever they are not necessary.

The only widely known results on the semantic strength of higher order logics in finite models are basically just extensions of the Fagin-Stockmeyer correspondence between the the prenex fragments of SO and fragments of the polynomial hierarchy. To state them, define, by induction on d, the d-fold exponential function $\exp_d$: $\exp_0$ is the identity function on the natural numbers and $\exp_{d+1}(n)$ is $2^{\exp_d(n)}$.

Let the complexity class dNEXP be $\bigcup_{k\in\omega} \mathsf{NTIME}(\exp_d(n^k))$. The classes dEXP, $\mathsf{dEXPSPACE}$, and $\mathsf{d}\Sigma_m\mathsf{EXP}$ are defined analogously ($\mathsf{d}\Sigma_m\mathsf{EXP}$ is defined by bounds on $\Sigma_m\mathsf{TIME}$; note that $\mathsf{d}\Sigma_1\mathsf{EXP}$ is dNEXP for any d). The dNEXP *hierarchy* consists of $\mathsf{d}\Sigma_m\mathsf{EXP}$ for all $m \geqslant 1$.

Example. $\mathsf{0NEXP}$ is simply NP, and $\mathsf{1NEXP}$ is NEXP. Similarly, $\mathsf{0EXP}$ is P, $\mathsf{1EXP}$ is EXP, $\mathsf{0EXPSPACE}$ is PSPACE and $\mathsf{1EXPSPACE}$ is $\mathsf{EXPSPACE}$.

The $\mathsf{0NEXP}$ hierarchy is PH. The $\mathsf{1NEXP}$ hierarchy is known as the *(full) weak exponential hierarchy* and is usually denoted by EXPH. Its levels are denoted by Σ_m^{exp}.

Let the class $\mathsf{ELEMENTARY}$ consist of problems solvable using resources bounded by $\exp_d$ for some d (observe that by the standard relations between bounds on different resources — see [BDG95] — it does not matter whether we take "resources" to mean DTIME or NTIME, DSPACE etc.). $\mathsf{ELEMENTARY}$ is the union of the dNEXP hierarchies. The following proposition summarizes the relationships between higher order logics and complexity classes:

Proposition 2.2. *For any $d, m \geqslant 1$:*

(a) Σ_m^d captures $(\mathsf{d}-1)\Sigma_m\mathsf{EXP}$;

(b) $\mathcal{L}^{d+1}$ captures the $(\mathsf{d}-1)\mathsf{NEXP}$ hierarchy;

(c) $\mathcal{L}^{\omega}$ captures $\mathsf{ELEMENTARY}$.

Remark. Proposition 2.2, which seems to be considered a part of the folklore, has had a strange history. Part (c) was shown by Bennett in his 1962 thesis ([Ben62]). As reported in [Bör84], the case $m = 1$ of part (a) was shown by Christen in his 1974 thesis ([Chr74]). Part (b) is stated, albeit without proof, in theorem 50 of [Lei87] (and it should be noted that at least the case $d = 2$, stated separately as theorem 48 of [Lei87], is based on a mistaken definition of the "full exponential time hierarchy").

Note, incidentally, that proposition 2.2 follows easily from theorem 2.5 in the next section of this chapter. A characterization of the expressive power of the Σ^d_m classes in terms of oracle machines instead of alternating machines, which could also be used to obtain proposition 2.2, has recently been given in [HT03].

2.2 A refinement of the prenex classes: $[\Sigma^d_m]^{\leqslant k}$

In this section we introduce and discuss, for each $m \geqslant 1$ and $d \geqslant 2$, certain syntactically defined subclasses of Σ^d_m.

We need to generalize the notion of arity of a second order variable (or type) to higher orders. Define the *basic arity* (ba) of a type τ of order $\geqslant 2$ in the following way:

Definition 2.3. If $\mathrm{order}(\tau) = 2$, then $\mathrm{ba}(\tau) = \mathrm{arity}(\tau)$. If $\mathrm{order}(\tau) > 2$ and $\tau = (\tau_1 \dots \tau_k)$, then $\mathrm{ba}(\tau) = \mathrm{ba}(\tau_{i_0})$, where τ_{i_0} has maximal basic arity among the τ_i of maximal order (i.e. of order equal to $\mathrm{order}(\tau) - 1$).

Example. For $\tau = (((\iota))\,((\iota\iota))\,(\iota\iota)\,\iota\iota\iota)$, $\mathrm{ba}(\tau) = 2$.

The basic arity of a variable equals the basic arity of its type.

Definition 2.4. For $d \geqslant 2$, $m, k \geqslant 1$, $[\Sigma^d_m]^{\leqslant k}$ is the class of Σ^d_m formulae in which the $(d+1)$-st and d-th order variables have basic arity at most k.

The definition of $[\Sigma^d_m]^{\leqslant k}$ may seem awkward. Why do we restrict the basic arity of variables of the two highest orders, instead of only the highest order — or perhaps all orders?

As for the latter possibility, the notion it yields is essentially just the one we have defined. Indeed, a definition of $[\Sigma^d_m]^{\leqslant k}$ where we required all the variables to have basic arity at most k would give a class with the same expressive power (at least with respect to sentences). This is because a second order relation R of some high arity r can be "coded" by a third order relation $\mathbf{R}$ of type

$$\underbrace{((\iota)\dots(\iota))}_{r}$$

(and thus of basic arity 1): just consider the $\mathbf{R}$, consisting exclusively of tuples of singletons, such that the tuple of singletons $(\{x_1\}, \dots, \{x_r\})$ is in $\mathbf{R}$ if and only if $(x_1, \dots, x_r)$ is in R. It is quite straightforward to transfer

this coding to higher orders. Given a Σ_m^d sentence, we can use the coding to obtain an equivalent sentence not containing low order variables with high basic arity. However, to keep the new formula in Σ_m^d, we may only apply that trick to variables of order up to $d-1$, as eliminating d-th order variables could generate extra alternations of $(d+1)$-st order quantifier blocks.

We defer the justification of our choice to restrict two orders rather than one until after the proof of the next theorem. The theorem shows that the $[\Sigma_m^d]^{\leqslant k}$ classes have natural complexity-theoretical counterparts.

Theorem 2.5. *For any $d \geqslant 2$, $m, k \geqslant 1$, $[\Sigma_m^d]^{\leqslant k}$ captures the complexity class $\bigcup_{c\in\omega} \Sigma_m \mathsf{TIME}(\exp_{d-1}(cn^k))$.*

Proof. ($\subseteq$) Fix k. A simple proof by induction on d shows that for any d and any type τ such that $2 \leqslant \mathrm{order}(\tau) \leqslant d+1$, $\mathrm{ba}(\tau) \leqslant k$, the number of objects of type τ over $\{0,\ldots,n-1\}$ is bounded by $\exp_d(cn^k)$ for some c.

Moreover, there is a natural representation of objects of type τ as binary strings of length at most $\exp_{d-1}(cn^k)$. This representation is also defined inductively. For the induction base, use the usual binary coding of elements of $\{0,\ldots,n-1\}$ and the usual coding of k-ary relations over $\{0,\ldots,n-1\}$ as strings of length n^k. For $\tau = (\tau_1 \ldots \tau_l)$, assume that we have already defined a representation of objects of type τ_i $(i = 1,\ldots,l)$ as binary strings. This induces a lexicographic ordering on the set of l-tuples $(R_1,\ldots,R_l)$ such that for all i, R_i is an object of type τ_i. An object $\mathbf{R}$ of type τ can therefore be represented by the string $s_{\mathbf{R}}$ such that the r-th symbol of $s_{\mathbf{R}}$ is 1 if and only if the r-th tuple $(R_1,\ldots,R_l)$ is in $\mathbf{R}$. Again by induction, one may prove that the lengths of the representing strings are as claimed.

Now fix $d \geqslant 2, m$, and let φ be a $[\Sigma_m^d]^{\leqslant k}$ sentence (in prenex form, with the $(d+1)$-st order quantifiers in front as required). To check whether φ holds in a model $\mathbf{M}$, a Σ_m machine T^φ does the following. Starting in an existential state, it guesses relations corresponding to the variables in the initial block of existential quantifiers in φ (this amounts to writing down a fixed number of binary strings of length bounded by $\exp_{d-1}(cM^k)$ for some c). T^φ then switches to a universal state and guesses relations corresponding to the first block of universal quantifiers, switches back to existential, and so on. Once the witnesses for all the $(d+1)$-st order quantifiers have been guessed, it remains to deterministically check the truth of the quantifier-free part of φ for all possible choices of relations corresponding to the d-th and lower order quantifiers. The number of relations over $\mathbf{M}$ corresponding to one such quantifier is bounded by $\exp_{d-1}(cM^k)$ for some c (here we need the

assumption that also the d-th order variables in φ have basic arity bounded by k). Since for $d \geqslant 2$, $\exp_{d-1}(c_1 n^k) \cdot \exp_{d-1}(c_2 n^k) \leqslant \exp_{d-1}((c_1 + c_2)n^k)$, the number of all possible choices, and hence the running time, obeys the required time bound.

($\supseteq$) This is a standard variation on the theme of the proof of Fagin's theorem (see e.g. [Imm99]). Let T be a Σ_m machine which works in time bounded by $\exp_{d-1}(cn^k)$. Let us list the possible contents of a tape cell during a run of T as $\gamma_1, \ldots, \gamma_r$ (the content of a cell is either a symbol from the machine alphabet, if the tape head is not on the cell, or an ordered pair "current state of T, machine alphabet symbol", if the head is on the cell).

For any given computation path of T, we construct a matrix of size $\exp_{d-1}(cn^k) \times \exp_{d-1}(cn^k)$ whose rows are indexed by points in time (t) and whose columns are indexed by units of T's tape space (s). The symbol γ_i appears in position (t, s) of the matrix if and only if γ_i is the content of cell s at time t along the given computation path. Observe that for an input of size n, we can let both t and s vary between 1 and $\exp_{d-1}(cn^k)$, since T is $\exp_{d-1}(cn^k)$ time-bounded and therefore has no chance to use more than $\exp_{d-1}(cn^k)$ tape cells[2]. This means that we can code both t and s by relations of type

$$\tau = \overbrace{(\ldots(}^{d-2}\overbrace{\underbrace{(\iota \ldots \iota)}_{k} \ldots \underbrace{(\iota \ldots \iota)}_{k}}^{c}\overbrace{)\ldots)}^{d-2},$$

since there are exactly $\exp_{d-1}(cn^k)$ objects of type τ over a universe of size n. Observe that $\mathrm{order}(\tau) = d$, $\mathrm{ba}(\tau) = k$.

We now introduce for each γ_i a relation variable $\mathbf{R}_i$ of type $(\tau\tau)$ for which, given a computation path of T, $\mathbf{R}_i(t, s)$ is to hold whenever γ_i is the symbol in position (t, s) of the matrix corresponding to this computation path (more precisely, we need m separate copies of the variable $\mathbf{R}_i$; which copy is used depends on how many alternations between existential and universal states of T have occurred before time t). Note that $\mathrm{order}(\mathbf{R}_i) = d + 1$, $\mathrm{ba}(\mathbf{R}_i) = k$. The $[\Sigma_m^d]^{\leqslant k}$ sentence which defines the class of models accepted by T has m alternating blocks of $(d + 1)$-st order quantifiers for the variables $\mathbf{R}_i$ (there

[2]Actually, that is not fully accurate, as we allow multi-tape machines, and an l-tape machine may use up to $l \cdot f$ tape cells in time f. One possible way of dealing with this bothersome technicality is to use the linear speed-up theorem to lower the running time by an arbitrary constant factor, for example l; this can be done without increasing the number of tapes. Another — is simply to take $c := c + 1$.

is one copy of each $\mathbf{R}_i$ in each block), followed by a formula in which no quantifiers of order greater than d appear, and which states that the input is appropriate and that the computation proceeds according to the transition function of T. □

Remark. Theorem 2.5 holds also over models without built-in arithmetic or even ordering, as each of the $[\Sigma^d_m]^{\leqslant k}$ classes can easily define an ordering (even for $k = 1$, as long as $d \geqslant 2$).

Example. It follows directly from the above theorem that the class NE is captured by $[\Sigma^2_1]^{\leqslant 1}$.

We may now return to the issue of justifying our definition of $[\Sigma^d_m]^{\leqslant k}$.

One consequence of theorem 2.5 is that for $d \geqslant 2$, the $[\Sigma^d_m]^{\leqslant k}$ classes form a natural hierarchy within Σ^d_m. On the one hand, each Σ^d_m-expressible property of models can be defined by a $[\Sigma^d_m]^{\leqslant k}$ sentence for some k. On the other hand, it follows from the time hierarchy theorems that $[\Sigma^d_m]^{\leqslant k}$ is strictly less expressive than $[\Sigma^d_m]^{\leqslant k+1}$.

Had we defined $[\Sigma^d_m]^{\leqslant k}$ by limiting just the basic arity of variables of maximal order, we would not be able to use time hierarchy results to obtain strict hierarchicity. This is because the proof of theorem 2.5 would break down: in the ($\subseteq$) direction, we need to loop through all relations corresponding to the d-th and lower order quantifiers in our sentence φ in deterministic time $\exp_{d-1}(cn^k)$. But if φ contains d-th order quantifiers of high basic arity, this may be impossible, as even the simplest d-th order type of basic arity $k+1$, i.e.

$$\overbrace{(\ldots(}^{d-1}\underbrace{\iota\ldots\iota}_{k+1}\overbrace{)\ldots)}^{d-1},$$

contains, over a universe of size n, as many as $\exp_{d-1}(n^{k+1})$ relations.

Notice that this is essentially the same problem as the one which prevents us from showing that Σ^1_1 sentences with relation variables of arity up to k capture $\mathsf{NTIME}(n^k)$: there, the number of tuples corresponding to $k+1$ first order quantifiers is n^{k+1}.

2.3 Closures of prenex classes under quantification of lower order

As already discussed, the expressive power of higher order prenex classes is generally known: Σ_m^d captures $(\mathsf{d}-1)\Sigma_m\mathsf{EXP}$. But the prenex classes are not the only ones deserving study. What happens, for example, when we close a higher order prenex class under boolean connectives and quantification of some lower order?

Define $Q_r(\Sigma_m^d)$ as the closure of Σ_m^d under connectives and quantification of r-th or lower order (obviously, this is interesting only for $r \leqslant d$).

The strength of $Q_1(\Sigma_m^d)$, which could also be denoted $FO(\Sigma_m^d)$, is known: it follows from theorem 4.9 and corollary 5.3 of [Got97] that $Q_1(\Sigma_m^d)$ captures $\mathsf{L}^{(\mathsf{d}-1)\Sigma_m\mathsf{EXP}}$. We will show that this result has a natural generalization: closures under r-th order quantification capture relativized $(\mathsf{r}-2)\mathsf{EXPSPACE}$ classes (where we use the convention that $(-1)\mathsf{EXPSPACE}$ is L).

Theorem 2.6. *For any $r, m, d \geqslant 1$, $r \leqslant d$, $Q_r(\Sigma_m^d)$ captures the complexity class $(\mathsf{r}-2)\mathsf{EXPSPACE}^{(\mathsf{d}-\mathsf{r})\Sigma_m\mathsf{EXP}}$.*

By the results of [Got97] mentioned above, we only need to deal with the case where $r \geqslant 2$, i.e. where we close Σ_m^d under at least second order quantification. We start with the more obvious direction, that a $Q_r(\Sigma_m^d)$-definable class of models can be recognized in $(\mathsf{r}-2)\mathsf{EXPSPACE}$ with a $(\mathsf{d}-\mathsf{r})\Sigma_m\mathsf{EXP}$ oracle. Let φ be a $Q_r(\Sigma_m^d)$-sentence over vocabulary σ. By standard arguments, we may assume w.l.o.g. that φ is

$$Q_1\mathbf{R}_1 \ldots Q_s\mathbf{R}_s\, \alpha(\psi_1, \ldots, \psi_t),$$

where the Q_i's are quantifiers, the R_i's are variables of order at most r, $\alpha(p_1, \ldots, p_t)$ is a propositional formula, and the ψ_i's are Σ_m^d formulae over σ. Note that, in general, the ψ_i's do not have to be sentences: they may contain free variables from among $\mathbf{R}_1, \ldots, \mathbf{R}_s$.

Let τ_i be the type of $\mathbf{R}_i$, for $i = 1, \ldots, s$. Since $\mathrm{order}(\mathbf{R}_i) \leqslant r$, then, as in the proof of theorem 2.5, objects of type τ_i over a universe of size n may be represented as binary strings of length at most $\exp_{r-2}(n^{k_i})$, for some k_i. Taking k to be the maximum of the k_i's, we see that given a σ-model $\mathbf{M}$ as input, we can loop through all possible choices of the $\mathbf{R}_i$'s over $\mathbf{M}$ deterministically in space $O(\exp_{r-2}(M^k))$.

Thus, to see that $MOD(\varphi) \in (\mathsf{r}-2)\mathsf{EXPSPACE}^{(\mathsf{d}-\mathsf{r})\Sigma_m\mathsf{EXP}}$ it remains to check that for a fixed choice of the $\mathbf{R}_i$'s and for any index $j = 1, \ldots, t$, a $(\mathsf{r}-2)\mathsf{EXPSPACE}$ machine can test whether

$$\mathbf{M} \models \psi_j(\mathbf{R}_1, \ldots, \mathbf{R}_s)$$

by querying a $(\mathsf{d}-\mathsf{r})\Sigma_m\mathsf{EXP}$ oracle. The formula ψ_j is Σ_m^d, so the time needed to evaluate in on a Σ_m machine might be up to $(d-1)$-fold exponential in M (even though ψ_j contains free variables, their order is lower than d+1, and so the proof of part ($\subseteq$) of theorem 2.5 works to show that $(d-1)$-fold exponential time is enough). Fortunately, we may apply the well-known technique of padding, that is, adding long easily recognizable strings (pads) to the inputs. A space-bounded oracle machine can write on its oracle tape some words which are exponentially long w.r.t. the space bound, say, a string of all zeroes. So, instead of querying $(\mathbf{M}, \mathbf{R}_1, \ldots, \mathbf{R}_s)$ to a $(\mathsf{d}-\mathsf{1})\Sigma_m\mathsf{EXP}$ oracle for ψ_j, our $(\mathsf{r}-2)\mathsf{EXPSPACE}$ machine queries

$$(\mathbf{M}, \mathbf{R}_1, \ldots, \mathbf{R}_s)^\frown 0^{\exp_{r-1}(M^k)}$$

(the code for $\mathbf{M}$ expanded by the $\mathbf{R}_i$'s concatenated with a long string of zeroes) to an oracle which simply disregards the $\exp_{r-1}(M^k)$-bit long pad and evaluates ψ_j. Since it took a Σ_m machine $(d-1)$-fold exponential time to check ψ_j, the pad will allow it to use just $((d-1)-(r-1))$-fold, i.e. $(d-r)$-fold, exponential time, as required.

One direction of theorem 2.6 is thus proved. The other, though apparently not immediately obvious, turns out to follow easily from two lemmas, both of which are actually rather straightforward adaptations of results already known from the literature.

The first lemma, a simple variant of theorems proved by L. Hemaspaandra (then Hemachandra) in [Hem89] and E. Hemaspaandra in [Hem94], shows that $\mathsf{rEXPSPACE}^{\mathsf{d}\Sigma_m\mathsf{EXP}}$ is, in fact, equal to a $\mathsf{DTIME}^{\Sigma_m\mathsf{TIME}}$ class which could at first sight seem smaller.

Lemma 2.7. *For any $d, r \geqslant 0, m \geqslant 1$,* $\mathsf{rEXPSPACE}^{\mathsf{d}\Sigma_m\mathsf{EXP}} = \mathsf{rEXP}^{(\mathsf{d}+1)\Sigma_m\mathsf{EXP}}$.

Proof. ($\supseteq$) This is immediate, by the fact that an rEXP-machine is, in particular, an $\mathsf{rEXPSPACE}$-machine, while an $\mathsf{rEXPSPACE}$-machine may pad its queries to lower the complexity of its oracle language by one exponent in comparison to that of an rEXP-machine.

($\subseteq$) In [Hem89], L. Hemaspaandra used census techniques to prove that $\mathsf{P}^{\mathsf{NE}} = \mathsf{NP}^{\mathsf{NE}}$. Later, E. Hemaspaandra noted in [Hem94] that a "straightforward application" of these techniques gives also

$$\mathsf{DSPACE}(f(n))^{\mathsf{NTIME}(g(n))} \subseteq \mathsf{DTIME}(poly(f(n)))^{\mathsf{NTIME}(poly(g(2^n)))},$$

for any time-constructible f and any g which is at least linear. This immediately yields our thesis for $m = 1$ and any r, d. In [Hem94], the details of the "straightforward application" are not spelled out. All we need to do is to sketch them in order to check that the techniques extend also to Σ_m oracle classes instead of NTIME oracle classes. The sketch follows.

Assume that L is a language recognized by the deterministic oracle machine T which uses up to $\exp_r(n^k)$ cells of space and has the language $N \in \Sigma_m\mathsf{TIME}(\exp_d(n^l))$ as its oracle.

By standard arguments, we may assume w.l.o.g. that T runs for at most $\exp_{r+1}(n^k)$ steps (and that it has both a clock and a space counter which prevent it from exceeding the time and space bounds). Indeed: if we neglect the oracle tape, a configuration of T is determined by a binary string representing the state, the positions of the tape heads, and the contents of the worktapes. Altogether, the string will have length $O(\exp_r(n^k))$. There are only $2^{O(\exp_r(n^k))}$ strings of such length, so there are at most $2^{O(\exp_r(n^k))}$ configurations of T which differ somewhere else than just on the oracle tape. Moreover, if T ever enters a configuration in which it has already been save perhaps for the contents of the oracle tape, the only way it can avoid entering an infinite loop is if the oracle query it is currently writing down gets a different answer than the first query asked after T reached this configuration previously (recall that the oracle tape is write-only). But since there are only two possible answers to oracle queries, YES and NO, reaching the configuration a third time will inevitably result in a loop. So, T cannot run for more than $2^{O(\exp_r(n^k))}$ steps, or else it will loop and never end its computation. Possibly by increasing the original k, we can get rid of the $O(\cdot)$ in $2^{O(\exp_r(n^k))}$, obtaining an $\exp_{r+1}(n^k)$ time bound.

We can now define a $(\mathsf{d}+\mathsf{1})\Sigma_m\mathsf{EXP}$ language N_T such that L can be recognized in deterministic r-fold exponential time with N_T as the oracle.

N_T is defined as the set of triples of the form $\langle v, x, c\rangle$, where c is a word of length $\Theta(\exp_r(|x|^k))$ coding a number $\leqslant O(\exp_{r+1}(|x|^k))$ — in binary notation with leading zeroes, for example — for which there exists a set of strings $C, card(C) = c$, so that:

1. $C \subseteq N$,

2. for any $w \in C$, there is a possible genuine configuration[3] ρ of T on input x such that w is the first query T will ask if started in configuration ρ,

3. if $v = 1$, then the simulation of T where a query q is answered YES iff $q \in C$ ends in an accepting state.

One may check that N_T is in $(\mathsf{d}+1)\Sigma_m\mathsf{EXP}$. Briefly: since there are no more than $O(\exp_{r+1}(|x|^k))$ possible genuine configurations of T on input x, and each of these may lead to at most one query, of length at most $\exp_{r+1}(|x|^k)$, the whole set C may be guessed as an $O(\exp_{r+1}(|x|^k))$-long string; but that is just exponential in the length of c. Checking $C \subseteq N$ is in $(\mathsf{d}+\mathsf{r}+1)\Sigma_m\mathsf{EXP}$ w.r.t. x — hence in $(\mathsf{d}+1)\Sigma_m\mathsf{EXP}$ once the length of c is counted in — by the fact that N itself is in $\mathsf{d}\Sigma_m\mathsf{EXP}$ and there are $(r+1)$-fold exponentially many strings to check, and by the closure of any $\mathsf{b}\Sigma_m\mathsf{EXP}$ class under conjunctions. Similar arguments show that checking the remaining two conditions can be performed in $(\mathsf{d}+1)\Sigma_m\mathsf{EXP}$.

Now, whenever $\langle v, x, c\rangle$ is in N_T, this means that there are at least c strings in N which may be asked to the oracle by T after starting in a possible genuine configuration on input x. An rEXP-machine can establish the maximal such c — c_0, say — by binary search, asking $O(\exp_r(|x|^k))$ queries of the form $\langle 0, x, c\rangle$. The only C corresponding to c_0 is then exactly the set of *all* possible queries which could be asked during a run of T on x and get a positive answer. So, once c_0 is determined, the rEXP-machine need just query $\langle 1, x, c_0\rangle$ to N_T: by clause 3. in the definition of N_T, the answer will amount to whether T accepts x. □

In particular then, the class $(\mathsf{r}-2)\mathsf{EXPSPACE}^{(\mathsf{d}-\mathsf{r})\Sigma_m\mathsf{EXP}}$ is exactly the same as $(\mathsf{r}-2)\mathsf{EXP}^{(\mathsf{d}-\mathsf{r}+1)\Sigma_m\mathsf{EXP}}$. Our second lemma now says that any class of models in $(\mathsf{r}-2)\mathsf{EXP}^{(\mathsf{d}-\mathsf{r}+1)\Sigma_m\mathsf{EXP}}$ can be defined by a $Q_r(\Sigma_m^d)$-sentence — moreover, by a $Q_r(\Sigma_m^d)$-sentence in a special normal form. The normal form

[3]By *possible* genuine configuration (of T on input x), we mean any tuple

(state of T, $\exp_{r-1}(|x|^k)$-bit strings to record the positions of the tape heads, $\exp_r(|x|^k)$-bit strings to record the worktape contents),

regardless of whether such a configuration may at all appear in the computation of T on x. Recall that a genuine configuration is one with an empty oracle tape, and note that the strings needed to represent such configurations are just $O(\exp_r(|x|^k))$ bits long.

is an analogue of a normal form obtained for logics capturing relativized logarithmic space classes by I. Stewart (theorem 3.3.1 and corollary 3.3.1 of [Ste93]) and G. Gottlob (corollary 5.3 of [Got97]), and dubbed the *Stewart normal form* (SNF) in [Got97].

Lemma 2.8. *Let σ be a vocabulary and let $r \geqslant 1$. Then any class of σ-models in $(r-2)\mathsf{EXP}^{(d-r+1)\Sigma_m\mathsf{EXP}}$ can be defined by a sentence of the form*

$$\exists \mathbf{R}\, (\varphi(\mathbf{R}) \,\&\, \neg\psi(\mathbf{R})),$$

where $\mathbf{R}$ is an r-th order variable, while φ and ψ are Σ^d_m formulae with $\mathbf{R}$ as the only free variable.

Since the proof is very similar not only to the one given by Gottlob, but also to the proof of our lemma 4.7 (which gives yet another variant of the SNF), we relegate it to an appendix.

By lemmas 2.7 and 2.8, any $(\mathsf{r}-2)\mathsf{EXPSPACE}^{(\mathsf{d}-\mathsf{r})\Sigma_m\mathsf{EXP}}$ class of models is indeed definable by a $Q_r(\Sigma^d_m)$-sentence, which completes the proof of theorem 2.6.

Additionally, we obtain the following corollary (where the case of $r = 1$ follows from [Got97]):

Corollary 2.9. *For any $r, m, d \geqslant 1$, $r \leqslant d$, every $Q_r(\Sigma^d_m)$-sentence is equivalent to a $Q_r(\Sigma^d_m)$-sentence in Stewart Normal Form:*

$$\exists \mathbf{R}\, (\varphi(\mathbf{R}) \,\&\, \neg\psi(\mathbf{R})),$$

where $\mathbf{R}$ is an r-th order variable, while φ and ψ are Σ^d_m formulae with $\mathbf{R}$ as the only free variable.

Remark. Just like theorem 2.5 (cf. the remark right after the proof), for $r \geqslant 2$ theorem 2.6 holds even over models without built-in relations, since we can use second or higher order quantification to construct the relations we need.

This contrasts with the case of $r = 1$, i.e. Gottlob's original result. There, it is known that a built-in linear order is enough to get the theorem, but unknown whether it still holds if there are no built-in relations at all. Actually, there are some good reasons to think it does not hold (see [DGH98]).

Let us close the chapter by listing some particular cases of theorem 2.6:

$Q_2(\Sigma^2_1)$, the second order closure of existential third order logic, captures $\mathsf{PSPACE}^{\mathsf{NP}}$. By [Hem89] and [Hem94], this class equals the so-called *strong*

exponential hierarchy SEH, defined as $\mathsf{NE} \cup \mathsf{NP}^{\mathsf{NE}} \cup \mathsf{NP}^{\mathsf{NP}^{\mathsf{NE}}} \ldots$, and shown in [Hem89] to collapse to $\mathsf{P}^{\mathsf{NEXP}}$.

Moving on to fourth order logic, we clearly see the differences between closures under quantification of various orders: $Q_1(\Sigma_1^3)$ captures $\mathsf{L}^{\mathsf{2NEXP}}$, $Q_2(\Sigma_1^3)$ captures $\mathsf{PSPACE}^{\mathsf{NEXP}}$, finally, $Q_3(\Sigma_1^3)$ captures $\mathsf{EXPSPACE}^{\mathsf{NP}}$.

Chapter 3

Truth Definitions

One of the reasons why the basic task of finite model theory, proving that some logics are more expressive in the finite than others, appears so daunting, is the scarcity of available methods. In particular, it seems that many methods of classical model theory are useless in finite models.

Let us recall the main idea of the method of truth definitions, which clearly ranks among the most classical techniques for comparing the semantical strength of logics. Its essence lies in Tarski's famous theorem on the undefinability of truth. The theorem states, roughly speaking, that a logic $\mathcal{L}$ closed under some basic first order constructions, most notably negation, does not define arithmetical truth for itself, i.e. there is no $\mathcal{L}$-formula with one free variable which is true of (a Gödel number of) an $\mathcal{L}$-sentence exactly when the sentence itself is true (say, in the standard model of arithmetic). Now, if one proves that a logic $\mathcal{L}'$, known to be at least as expressive as $\mathcal{L}$, additionally defines truth for $\mathcal{L}$, then one has also shown — using the "method of truth definitions" — that $\mathcal{L}'$ is strictly more expressive than $\mathcal{L}$.

The classical method of truth definitions makes sense only in infinite models: the set of truths about any (possibly finite) model is always infinite, so it cannot be defined in a finite model. However, trying to adapt the method to the needs of finite model theory does not seem hopeless. Such an attempt was made by M. Mostowski, who introduced the notion of a *finite model truth definition* (*FM-truth definition*, for short) and proved, among other results, a finite version of Tarski's theorem.

The potential significance of FM-truth definitions as a tool in finite model theory remains difficult to assess. Due to some well-known limitations of straightforward diagonal arguments in finite model theory (or complexity

theory), it is unlikely that any of the hard open problems on whether two given logics are equivalent in finite models can be solved by showing that one of the logics has an FM-truth definition for the other. On the other hand, as illustrated in the next chapter, the concept of FM-truth definition definitely does have interesting applications.

The current chapter, meanwhile, contains proofs of some simple but useful general results on the behaviour of FM-truth definitions.

The first section of the chapter introduces the notion itself and recalls Mostowski's early results. In the second section, we go on to show that (perhaps somewhat surprisingly) the difficulty of defining FM-truth for a logic $\mathcal{L}$ depends only on the expressive power of $\mathcal{L}$, not on its syntax.

In section 3.3, we use this result to deduce that for a wide range of complexity classes C, given a logic $\mathcal{L}$ which captures C, there exist FM-truth definitions for $\mathcal{L}$ whose evaluation requires resources "barely above" C. This allows us to give exact complexity-theoretical characterizations of the classes of logics for which FM-truth is definable in one of the usual logics met in finite model theory (such as second order logic, its fragments, or fixed point logic).

Finally, we compare the idea of FM-truth definitions with an earlier, apparently related, concept of *combined complexity* introduced by Vardi.

3.1 Definitions and early results

The notion of FM-truth definition proposed in [Mos01] is an "asymptotic" one:

Definition 3.1. Let $\mathcal{L}$ be a logic and σ be a vocabulary. The σ-formula $Tr_{\mathcal{L},\sigma}(x)$, with x as the unique free variable, is an *FM-truth definition* for $\mathcal{L}$ over σ if and only if for every $\mathcal{L}$-sentence ψ of vocabulary σ,

$$\mathbf{M} \models \psi \equiv Tr_{\mathcal{L},\sigma}(\ulcorner \psi \urcorner)$$

holds for almost all σ-models $\mathbf{M}$.

Remark. Under our definition of logic, there are logics for which the notion of a formula with a free (first order) variable does not, strictly speaking, make sense. The formal way to deal with this problem is to think of such "formulae" as sentences of vocabulary $(\sigma + c)$, where c is a new individual

constant. The details of a suitable redefinition of "FM-truth definition" are straightforward.

So, if we have an FM-truth definition for $\mathcal{L}$, then for a given $\mathcal{L}$-sentence ψ, there is a number n such that in all models of cardinality greater than n the definition "knows" whether ψ is true or not. It will turn out that the "asymptoticity" of FM-truth definitions, i.e. the fact that we allow $Tr_{\mathcal{L},\sigma}(\ulcorner\psi\urcorner)$ to err in a finite number of models, has significant consequences.

If $\mathcal{L}, \mathcal{L}'$ are logics and there is an $\mathcal{L}'$-formula which is an FM-truth definition for $\mathcal{L}$ over vocabulary σ, we say that $\mathcal{L}'$ *defines FM-truth* for $\mathcal{L}$ over σ, and write $\mathcal{L} \ll_\sigma \mathcal{L}'$.

The finite version of Tarski's theorem is as follows:

Theorem 3.2 ([Mos01]; Tarski's theorem, finite version). *If $\mathcal{L}$ is a logic closed under first order quantification, forming conjunctions with first order formulae, and negation, then for any σ, $\mathcal{L} \not\ll_\sigma \mathcal{L}$.*

The theorem is, as in the classical case, an easy consequence of a diagonal lemma which states that for any formula with one free variable, there is a sentence which asserts of itself the property defined by the formula[1]. The finite version of the diagonal lemma is also formulated and proved in [Mos01].

The paper [Mos01] also contains a positive result. The standard proof that $\mathcal{L}^{d+1}$ has a truth definition (in the traditional sense) for $\mathcal{L}^d$ can be adapted to give:

Theorem 3.3 ([Mos01]). *For any d and σ, $\mathcal{L}^d \ll_\sigma \mathcal{L}^{d+2}$.*

Note that in order to define finite model truth for d-th order logic, we need to increase the order by 2 and not just by 1. The main obstacle to simply copying the classical proof, which increases the order by 1, is the lack of a pairing function in any particular finite model.

The problem whether $\mathcal{L}^d \ll_\sigma \mathcal{L}^{d+1}$ is still open, for any vocabulary σ. Actually, it follows from the results of section 3.3 that this problem is equivalent to a (presumably hard) problem in complexity theory, hence it seems unlikely that it will be solved soon.

[1] Again, in the general case the notion of "formula with one free variable" should be understood as "sentence of a vocabulary with one additional constant" — see the remark after definition 3.1.

3.2 Independence of syntax

Consider the following situation. Over a vocabulary σ, a logic $\mathcal{L}$ defines FM-truth for a logic $\mathcal{L}_1$. The logic $\mathcal{L}_2$, in turn, is known to be semantically contained in $\mathcal{L}_1$. Must it then hold that $\mathcal{L}$ defines FM-truth for $\mathcal{L}_2$ as well?

Although it may seem at first sight that the answer surely must be positive, there is a problem: the syntax of $\mathcal{L}_2$ might be extremely convoluted, and checking whether a $\mathcal{L}_2$-sentence is satisfied in a model might be much more difficult than in the case of $\mathcal{L}_1$. Additionally, even though for every $\mathcal{L}_2$-sentence there is an equivalent $\mathcal{L}_1$-sentence, there might be no effective (i.e. computable) way to find it!

Nevertheless, the answer *is* positive, under very modest assumptions about $\mathcal{L}$, and no assumptions whatsoever about the $\mathcal{L}_i$'s. Moreover, the proof is amazingly simple:

Theorem 3.4. *Let $\mathcal{L}_1$, $\mathcal{L}_2$ be logics, and let $\mathcal{L}$ be a logic closed under first order quantification and under taking conjunctions and disjunctions with first order formulae. In that case, for any σ, if $\mathcal{L}_1 \ll_\sigma \mathcal{L}$ and $\mathcal{L}_2 \leqslant_\sigma \mathcal{L}_1$, then $\mathcal{L}_2 \ll_\sigma \mathcal{L}$.*

Proof. The relation

"x_1 is an $\mathcal{L}_1$-sentence of vocabulary σ, x_2 is an $\mathcal{L}_2$-sentence of vocabulary σ, and x_1 is equivalent to x_2 in all σ-models"

is co-RE and therefore, by theorem 1.2, FM-representable. So, let it be FM-represented by the first order formula $\mathrm{eq}(x_1, x_2)$.

Let the $\mathcal{L}$-formula $Tr_{\mathcal{L}_1}(x_1)$ be an FM-truth definition for $\mathcal{L}_1$ over σ. Then the $\mathcal{L}$-formula

$$Tr_{\mathcal{L}_2}(x_2) := \exists x_1 \, (\mathrm{eq}(x_1, x_2) \,\&\, \forall y < x_1 \, \neg \mathrm{eq}(y, x_2) \,\&\, Tr_{\mathcal{L}_1}(x_1))$$

is an FM-truth definition for $\mathcal{L}_2$ over σ.

Indeed, let φ be an $\mathcal{L}_2$-sentence of vocabulary σ. There exists $\ulcorner \psi \urcorner$ such that $\ulcorner \psi \urcorner$ is the smallest (Gödel number of an) $\mathcal{L}_1$-sentence equivalent to φ. For all sufficiently large σ-models $\mathbf{M}$, it holds that

$$\mathbf{M} \models \mathrm{eq}(\ulcorner \psi \urcorner, \ulcorner \varphi \urcorner) \,\&\, \forall y < \ulcorner \psi \urcorner \, \neg \mathrm{eq}(y, \ulcorner \varphi \urcorner).$$

Similarly, for all sufficiently large $\mathbf{M}$, we have $\mathbf{M} \models Tr_{\mathcal{L}_1}(\ulcorner \psi \urcorner)$ iff ψ is true in $\mathbf{M}$. Thus, for all sufficiently large $\mathbf{M}$, $\mathbf{M} \models Tr_{\mathcal{L}_2}(\ulcorner \varphi \urcorner)$ iff φ is true in $\mathbf{M}$.

(Note that $\exists x_1\,(\mathrm{eq}(x_1, x_2)\,\&\,Tr_{\mathcal{L}_1}(x_1))$ would not have been sufficient, for two reasons. Firstly, eq may erroneously consider some $\mathcal{L}_1$-sentence equivalent to φ. Secondly, even if all sentences ψ satisfying $\mathrm{eq}(\ulcorner\psi\urcorner, \ulcorner\varphi\urcorner)$ are indeed equivalent to φ, $Tr_{\mathcal{L}_1}$ may be wrong about the truth value of some such ψ. Both situations may occur in arbitrarily large models: even though, as we increase the size of models, eq and $Tr_{\mathcal{L}_1}$ eventually become correct with respect to a particular ψ, new sentences ψ enter the models and the problem reappears). □

In other words, the question whether one logic defines FM-truth for another depends only on the semantical strength of the logics involved, and is utterly independent of the peculiarities of their syntax. As we will see in a moment, this fact has important consequences.

3.3 The complexity of defining FM-truth

Assume a logic $\mathcal{L}$ captures a complexity class C. No matter what C is, it is clearly possible to define, in some artificial logics, some artificial FM-truth definitions for $\mathcal{L}$ which will be very difficult to evaluate. A more interesting question is: how easy can an FM-truth definition for $\mathcal{L}$ be made?

It follows from the hierarchy theorems that if C is one of the usual complexity classes, the evaluation of an FM-truth definition for $\mathcal{L}$ must certainly require more resources than C itself permits. As a matter of fact, this is (almost) all that is needed — it turns out the evaluation of some FM-truth definitions for $\mathcal{L}$ requires resources "barely above" C.

Once we have theorem 3.4, the proof is basically an application of the simple idea used to prove the deterministic version of the hierarchy theorems.

Theorem 3.5. *Let $\mathcal{L}$ be a logic such that model checking for $\mathcal{L}$ over the vocabulary σ is in* $\mathsf{DSPACE}(f)$ *(resp.* $\mathsf{DTIME}(f), \mathsf{NTIME}(f), \Sigma_m\mathsf{TIME}(f)$*), where f is some space-constructible (resp. time-constructible) function. Let g be a space-constructible (resp. time-constructible) function such that $f = o(g)$ (resp. $f \cdot \log f = o(g)$). If the logic $\mathcal{L}'$, closed under first order quantification and under taking conjunctions and disjunctions with first order formulae, captures at least* $\mathsf{DSPACE}(g)$ *(resp.* $\mathsf{DTIME}(g), \mathsf{NTIME}(g), \Sigma_m\mathsf{TIME}(g)$*), then $\mathcal{L} \ll_\sigma \mathcal{L}'$.*

Proof. We elaborate only on the DSPACE case. By theorem 3.4, it is enough to find a logic $\mathcal{L}^*$ which captures at least $\mathsf{DSPACE}(f)$ such that $\mathcal{L}^* \ll_\sigma \mathcal{L}'$.

Let the set of $\mathcal{L}^*$-sentences consist of (the codes of) all deterministic Turing machines which work within space bounded by f (to ensure decidability of syntax, we may assume that the set of $\mathcal{L}^*$-sentences consists of (the codes of) all DTMs equipped with some standard device which forces the machine to reject whenever it tries to use more space than is allowed by f; this is possible because of the space-constructibility of f). The semantics of $\mathcal{L}^*$ is defined by positing that for any model $\mathbf{M}$ and machine T, $\mathbf{M} \models T$ if and only if T accepts $\mathbf{M}$. Obviously, $\mathcal{L}^*$ captures exactly $\mathsf{DSPACE}(f)$.

Consider the Turing machine $T_{\mathcal{L}^*}$ which on input $(\mathbf{M}, n)$, where $\mathbf{M}$ is a σ-model, checks whether n is of the form $\ulcorner T \urcorner$ for some $\mathcal{L}^*$-machine T, and rejects if it is not. Otherwise, $T_{\mathcal{L}^*}$ simulates T on $\mathbf{M}$ and accepts if and only if T does. Additionally, $T_{\mathcal{L}^*}$ uses a device which forces it to stop and reject whenever it tries to use more space than $g(M)$ (again, this is possible as g is space-constructible).

Clearly, $T_{\mathcal{L}^*}$ is a $\mathsf{DSPACE}(g)$-machine. $\mathcal{L}'$ captures at least $\mathsf{DSPACE}(g)$, so there exists an $\mathcal{L}'$-sentence $Tr_{\mathcal{L}^*}(c)$ in the vocabulary $(\sigma + c)$ (where $c \notin \sigma$) such that $(\mathbf{M}, n) \models Tr_{\mathcal{L}^*}(c)$ if and only if $T_{\mathcal{L}^*}$ accepts $(\mathbf{M}, n)$.

Moreover, for any $\mathcal{L}^*$-sentence T, the machine $T_{\mathcal{L}^*}$ will be able to complete its computation on $(\mathbf{M}, \ulcorner T \urcorner)$ for all sufficiently large $\mathbf{M}$ (as $f = o(g)$). It therefore follows that the $\mathcal{L}'$-formula $Tr_{\mathcal{L}^*}(x)$ is an FM-truth definition for $\mathcal{L}^*$ over σ.

The case of time-bounded machines is similar, except for the additional $\log f$ factor in the assumptions, introduced in order to allow $T_{\mathcal{L}^*}$ to simulate $\mathcal{L}^*$-machines with an arbitrarily large tape alphabet and number of tapes (see e.g. [BDG95]). □

Given theorem 3.5, it becomes straightforward to characterize the logics for which, say, the prenex fragments of second or higher order logic define FM-truth:

Theorem 3.6. *Let $\mathcal{L}$ be a logic and let σ be a vocabulary. For any $d \geqslant 1$, it holds that $\mathcal{L} \ll_\sigma \Sigma^d_m$ if and only if there exists a number k such that model checking for $\mathcal{L}$ over σ is in $\Sigma_m \mathsf{TIME}(\exp_{d-1}(n^k))$.*

Proof. For the "only if" direction, let φ be an $\mathcal{L}$-sentence of vocabulary σ and let $Tr_{\mathcal{L},\sigma}(x)$ be the Σ^d_m FM-truth definition for $\mathcal{L}$. By (a) of proposition 2.2, there is some k and some Σ_m machine T which checks if $(\mathbf{M}, \ulcorner \varphi \urcorner) \models Tr_{\mathcal{L},\sigma}(c)$ in time bounded by $\exp_{d-1}(M^k)$. But then the machine T^φ which, on input $\mathbf{M}$, writes down $\ulcorner \varphi \urcorner$ on its tape, simulates T on $(\mathbf{M}, \ulcorner \varphi \urcorner)$, and accepts if

and only if T does, is also a Σ_m machine and works in time bounded by $\exp_{d-1}(n^k)$. Moreover, T^φ determines the truth value of φ correctly in all sufficiently large models.

For the "if" direction, just observe that for any d and k it holds that $\exp_d(n^k) \cdot \log(\exp_d(n^k)) = o(\exp_d(n^{k+1}))$ — and apply theorem 3.5. □

This characterization has several corollaries:

Corollary 3.7. *For any σ and any $d, m \leqslant 1$, $\Sigma^d_m \not\ll_\sigma \Sigma^d_m$.*

Corollary 3.8. *For any logic $\mathcal{L}$, any vocabulary σ, and any $d \geqslant 2$, it holds that $\mathcal{L} \ll_\sigma \mathcal{L}^d$ if and only if there exist m, k such that model checking for $\mathcal{L}$ over σ is in $\Sigma_m\mathsf{TIME}(\exp_{d-2}(n^k))$. Similarly, $\mathcal{L} \ll_\sigma \mathcal{L}^\omega$ if and only if there exists d such that model checking for $\mathcal{L}$ over σ is in* dNEXP*.*

Example. In [MP96], Makowsky and Pnueli use a diagonal method somewhat akin to FM-truth definitions to prove a hierarchy theorem for fragments of second order logic. They define $AA(k, m)$ as the class of those second order formulae in prenex normal form (here it is *not* required that the second order quantifiers precede the first order quantifiers) in which no relational variables of arity greater than k appear, and the number of alternating quantifier blocks in the prefix (counting both first and second order quantifiers) does not exceed m. It is shown that $AA(k, m)$ is less expressive than the whole second order logic (actually, it is shown that $AA(1, m) < AA(3, m + 4)$ and $AA(k, m) < AA(k + 1, m + 4)$ for $k \geqslant 2$).

Observe that model checking for the "existential" fragment of $AA(k, m)$ is clearly in $\Sigma_m\mathsf{TIME}(n^k)$, and model checking for the "universal" fragment is in $\Pi_m\mathsf{TIME}(n^k)$. Together, model checking for $AA(k, m)$ is in $\Sigma_{m+1}\mathsf{TIME}(n^k)$, so for any vocabulary σ, $AA(k, m) \ll_\sigma \Sigma^1_{m+1}$ and hence $AA(k, m) < \Sigma^1_{m+1}$. Since over ordered models (or even over arbitrary models, by the normal form theorem of [EGG96]) $\Sigma^1_{m+1} \leqslant \bigcup_{l\in\omega} AA(l, m + 2)$, it follows that for any k, m there exists a c such that $AA(k, m) < AA(k + c, m + 2)$.

Corollary 3.9. *For all $d \geqslant 2$, $m, k \geqslant 1$, and any σ, $[\Sigma^d_m]^{\leqslant k} \ll_\sigma \Sigma^d_m$.*

Remark. Actually writing down an FM-truth definition for $[\Sigma^d_m]^{\leqslant k}$ in Σ^d_m is rather tedious task. We give an example of such a definition in appendix B.

Of course, using arguments entirely analogous to the proof of theorem 3.6, we can characterize the classes of logics for which FM-truth can be defined in logics capturing P, PSPACE, etc. For example[2]:

Theorem 3.10. *For any $\mathcal{L}$ and σ:*

1. *$\mathcal{L} \ll_\sigma LFP$ if and only if there exists k such that model checking for $\mathcal{L}$ over σ is in $\mathsf{DTIME}(n^k)$;*

2. *$\mathcal{L} \ll_\sigma PFP$ if and only if there exists k such that model checking for $\mathcal{L}$ over σ is in $\mathsf{DSPACE}(n^k)$.*

3.4 Truth definitions and combined complexity

In [Var82], Vardi distinguished three notions of complexity associated with a logic. The *data complexity* of $\mathcal{L}$ is the complexity of determining $\mathbf{M} \models \varphi$ for varying models $\mathbf{M}$ and fixed $\mathcal{L}$-sentences φ — in other words, the complexity class captured by $\mathcal{L}$. The *expression complexity* is defined dually, with the models fixed and the sentences varying. Finally, the *combined complexity* of $\mathcal{L}$ is the complexity of checking $\mathbf{M} \models \varphi$ where both the models and the sentences vary.

Combined complexity appears to be a very similar notion to FM-truth definitions. Thus, a comment on the relation between these two ideas seems to be in order.

Note the basic difference: the combined complexity of $\mathcal{L}$ is, at least over a fixed vocabulary σ, a fixed level of complexity — the complexity of a "universal algorithm" testing $\mathbf{M} \models \varphi$ correctly for *all* $\mathbf{M}$ and φ — whereas FM-truth definitions for $\mathcal{L}$ are entitled to "a finite number of mistakes per sentence" and hence, as discussed in the previous section, the complexity of their evaluation may vary.

Another difference is that the combined complexity of a logic depends in general not only on its expressive power, but also on the syntax. Nevertheless, already Vardi observed that for many of the usual logics, the combined

[2]For the following theorem, the exact definitions of the logics LFP and PFP are inessential, all that matters is the complexity classes they capture — see the relevant results stated at the end of the *Preliminaries*.

complexity is one exponential level higher than the complexity class captured by the logic: e.g., the combined complexity of Σ^1_1 is NEXP-complete. This phenomenon was thoroughly investigated by Gottlob, Leone, and Veith ([GLV99]), who identified some sufficient conditions for a logic to obey this pattern, gave numerous examples of logics which do, and an example of a logic which does not. Observe that our theorem 3.5 shows that for all the logics which do fit this pattern, there exist FM-truth definitions whose evaluation is much simpler than the combined complexity would suggest.

Below, we try to describe the relation of FM-truth definitions to combined complexity with some degree of precision. This is essentially very simple, save for the unfortunate necessity to introduce some more technical concepts.

We start by formulating the combined complexity of a logic $\mathcal{L}$ over a fixed vocabulary σ as the complexity of a certain class of finite models. To simplify things, assume that $\mathcal{L}$-sentences are words over $\{0,1\}$ (if they are words over some other B instead, use the translation f_B defined in the *Preliminaries*). Let σ^+ be the expansion of σ by two unary predicates U^1, P^1. An ordered pair $(\mathbf{M}, w)$, where $\mathbf{M}$ is a σ-model and $w \in \{0,1\}^*$, may now be thought of as the single σ^+-model $(\mathbf{M}+w)$ defined as follows. The universe of the model $(\mathbf{M}+w)$ is $M+\mathrm{lh}(w)$. $U^{(\mathbf{M}+w)} = M$; thus, U divides $(\mathbf{M}+w)$ into an $\mathbf{M}$-part $\{0,\ldots,M-1\}$ and a w-part $\{M,\ldots,M+\mathrm{lh}(w)-1\}$. The interpretations of the non-arithmetical σ-symbols agree with $\mathbf{M}$ over the $\mathbf{M}$-part and are empty over the w-part. The interpretation of P is empty over the $\mathbf{M}$-part, while for any $1 \leqslant k \leqslant \mathrm{lh}(w)$, $P(M+k-1)$ holds iff the k-th symbol in w is 1. The interpretation of the arithmetic is, of course, standard.

For a given logic $\mathcal{L}$ and vocabulary σ, we associate with $\mathcal{L}$ the following *combined complexity class*:

$$CC_{\mathcal{L},\sigma} = \{(\mathbf{M}+\varphi) : \mathbf{M} \text{ is a } \sigma\text{–model, } \varphi \text{ is an } \mathcal{L}\text{-sentence of vocabulary } \sigma, \text{ and } M \models \varphi\}.$$

Observe that the complexity of $CC_{\mathcal{L},\sigma}$ is exactly the combined complexity of $\mathcal{L}$ over σ.

We also need the idea of *first order reductions*. This is an important notion in descriptive complexity, but since it is not used anywhere else in the thesis, we shall not define it precisely. Suffice it to say that a k-ary first order reduction of a class $\mathcal{K}$ of σ-models to a class $\tilde{\mathcal{K}}$ of $\tilde{\sigma}$-models is a tuple I of first order formulae (of vocabulary σ) defining, in any σ-model $\mathbf{M}$, a $\tilde{\sigma}$-model $\mathcal{I}(\mathbf{M})$ whose universe is a (first order definable) subset of M^k. $\mathcal{I}$ must have

the further property that for any $\mathbf{M}$,

$$\mathbf{M} \in \mathcal{K} \text{ iff } \mathcal{I}(\mathbf{M}) \in \tilde{\mathcal{K}}.$$

If a first order reduction of $\mathcal{K}$ to $\tilde{\mathcal{K}}$ exists, $\mathcal{K}$ is said to be *first order reducible to* $\tilde{\mathcal{K}}$. A logic $\mathcal{L}$ is *closed under first order reductions* if any class of models which first order reducible to an $\mathcal{L}$-definable class is also $\mathcal{L}$-definable.

Now it becomes easy to show that if a (reasonable) logic $\mathcal{L}'$ expresses the combined complexity class $CC_{\mathcal{L},\sigma}$, then $\mathcal{L}'$ also defines FM-truth for $\mathcal{L}$ over σ. To this end, define the *canonical truth-definition class* $TD_{\mathcal{L},\sigma}$:

$$TD_{\mathcal{L},\sigma} = \{(\mathbf{M}, \ulcorner\varphi\urcorner) : \mathbf{M} \text{ is a } \sigma\text{-model, } \varphi \text{ is an } \mathcal{L}\text{-sentence of vocabulary } \sigma, \ulcorner\varphi\urcorner \in M, \text{ and } M \models \varphi\}.$$

Proposition 3.11. *For any logic $\mathcal{L}$ and any vocabulary σ, the class $TD_{\mathcal{L},\sigma}$ is first order reducible to $CC_{\mathcal{L},\sigma}$. Hence, for any logic $\mathcal{L}'$ which defines $CC_{\mathcal{L},\sigma}$ and is closed under first order reductions, it holds that $\mathcal{L} \ll_\sigma \mathcal{L}'$.*

Proof. The reduction is to yield $(\mathbf{M} + w)$ on input $(\mathbf{M}, \ulcorner w\urcorner)$. We take it to be binary: the universe of the model we build will consist of those pairs (b, n) (where $b, n < M$) for which either $b = 0$ (this will be the $\mathbf{M}$-part of $(\mathbf{M}+w)$) or $b = 1$ and $n < \mathrm{lh}(w)$ (this will be the w-part). To define $\mathrm{lh}(w)$ in terms of $\ulcorner w\urcorner$, we say that $\mathrm{lh}(w)$ is the greatest number l for which the $l+1$-st bit of the binary expansion of $\ulcorner w\urcorner$ is 1. Here we are using the well-known fact that the relation "the x-th bit of the binary expansion of y is 1" is first order definable using $+$ and $\times$ (see [Imm99]).

Also the interpretations of the σ^+-symbols in our model are easily first order definable. $P(0, n)$ never holds and $P(1, n)$ holds iff the $(n+1)$-st character in w is 1, in other words, if and only if the $(n+1)$-st bit of $\ulcorner w\urcorner$ is 1. □

So, in reasonably well-behaved cases, a logic strong enough to capture the combined complexity of $\mathcal{L}$ (i.e. define $CC_{\mathcal{L},\sigma}$) is also strong enough to define FM-truth for $\mathcal{L}$. By the results of the previous section, there is no chance for a converse to proposition 3.11. However, we may use the proposition and theorem 3.4 to summarize the connection between FM-truth definitions and combined complexity as follows:

Corollary 3.12. *Let $\mathcal{L}, \mathcal{L}'$ be logics such that $\mathcal{L}'$ is closed under first order reductions, and let σ be a vocabulary.*

(a) *If there exists any logic* $\tilde{\mathcal{L}}$ *satisfying* $\tilde{\mathcal{L}} \equiv_\sigma \mathcal{L}$ *for which* $\mathcal{L}'$ *defines* $CC_{\tilde{\mathcal{L}},\sigma}$*, then* $\mathcal{L} \ll_\sigma \mathcal{L}'$*.*

(b) *If* $\mathcal{L} \ll_\sigma \mathcal{L}'$*, then there exists a logic* $\tilde{\mathcal{L}}$ *such that* $\mathcal{L}'$ *defines* $CC_{\tilde{\mathcal{L}},\sigma}$*, and for any* $\mathcal{L}$*-definable class* K *of* σ*-models there is an* $\tilde{\mathcal{L}}$*-definable finite variant of* K*.*

Proof. (a) This follows directly from proposition 3.11 and theorem 3.4.

(b) Let $Tr_{\mathcal{L},\sigma}(x)$ be the FM-truth definition. Define the syntax of $\tilde{\mathcal{L}}$ to be the same as that of $\mathcal{L}$. For an $\mathcal{L}$-sentence φ viewed as an $\tilde{\mathcal{L}}$-sentence, define $\mathbf{M} \models \varphi$ to hold if and only if $\ulcorner\varphi\urcorner \in \mathbf{M}$ and $\mathbf{M} \models Tr_{\mathcal{L},\sigma}(\ulcorner\varphi\urcorner)$. It is straightforward to verify that $CC_{\tilde{\mathcal{L}},\sigma}$ is first order reducible to the class $\{(\mathbf{M}, c) : \mathbf{M} \models Tr_{\mathcal{L},\sigma}(c)\}$. □

Part (b) of the corollary is clearly not satisfactory because the logic $\tilde{\mathcal{L}}$ we construct in the proof is very artificial and badly behaved: it is not even closed under taking finite variants of definable classes of models. It might be interesting to see whether for natural logics $\mathcal{L}$, part (b) still holds if we require $\tilde{\mathcal{L}}$ to satisfy some minimal natural assumptions.

Chapter 4

Applications

This chapter presents some applications of the framework developed in the thesis to computational complexity theory. Our main interest lies in the *bounded query* fragments of the polynomial hierarchy. Define $\mathsf{P}^{\mathsf{NP}[n^r]}$ to be the class of problems which can be solved by a deterministic polynomial-time oracle machine which *makes at most* n^r *queries* to an NP oracle on inputs of size n. Note that $\mathsf{P}^{\mathsf{NP}} = \bigcup_{r\in\omega} \mathsf{P}^{\mathsf{NP}[n^r]}$. Naturally, $\mathsf{P}^{\mathsf{C}[n^r]}$ can be defined analogously for any class C, in particular for $\mathsf{C} = \Sigma_i^p$.

As noted in the *Introduction*, the $\mathsf{P}^{\mathsf{NP}[n^r]}$ classes are the largest fragments of PH known to be smaller than NEXP. The problem whether $\mathsf{P}^{\mathsf{NP}} = \mathsf{NEXP}$ is open and has contradictory relativizations (i.e., there is both a language A such that $\mathsf{P}^{\mathsf{NP}^A} \subsetneq \mathsf{NEXP}^A$ and a language B such that $\mathsf{P}^{\mathsf{NP}^B} = \mathsf{NEXP}^B$), so it is likely to be very difficult[1]. The theorems proved in this chapter may by viewed as an explanation of this initially surprising difference between P^{NP} and $\mathsf{P}^{\mathsf{NP}[n^r]}$ in terms of the structure of the polynomial hierarchy.

The chapter is divided into four sections. In section 4.1, we discuss a suitable notion of universal language for a complexity class, introduced in [NIR03] under the name *weak universal language*. We point out a close affinity between universal languages for a given class and FM-truth definitions for a logic capturing that class.

In the next two sections, we exploit this affinity to prove three results of the form: "if C (an appropriate fragment of PH) contains a universal language

[1]It is conventional wisdom in complexity theory that problems of the form "$\mathsf{C} = \mathsf{D}$?" which have contradictory relativizations are very hard, especially if it is conjectured that $\mathsf{C} \neq \mathsf{D}$. This view has undergone some criticism ([HCC+92], [For94]), but it still seems to provide a reasonable criterion for "very-hardness" in most typical cases.

for NP, then it also contains one for $\mathsf{P}^{\mathsf{NP}[n^r]}$". Since the hierarchy theorems imply that none of the usual complexity classes has a universal language for itself, any of these results is enough to get $\mathsf{P}^{\mathsf{NP}[n^r]} \neq \mathsf{NEXP}$. Indeed, if NEXP, which has a universal language for NP, were to equal $\mathsf{P}^{\mathsf{NP}[n^r]}$, and thus C, it would also have a universal language for $\mathsf{P}^{\mathsf{NP}[n^r]}$. Together with the equality $\mathsf{P}^{\mathsf{NP}[n^r]} = \mathsf{NEXP}$, this would entail that NEXP has a universal language for itself, which is known to be false.

The fourth and final section of the chapter discusses some generalizations and modifications of the results proved in sections 4.2 and 4.3.

4.1 Universal languages for complexity classes

Definition 4.1. $\mathcal{U}$ is a *universal language* for a complexity class $\mathcal{C}$ if

$$\forall L \in \mathcal{C}\ \exists x \in \Sigma^*\ \forall y \in \Sigma^*\ (y \in L \iff \langle x, y\rangle \in \mathcal{U}).$$

In other words, $\mathcal{U}$ is universal for $\mathcal{C}$ if for any language $L \in \mathcal{C}$ there is a word $x \in \Sigma^*$ such that $\mathcal{U}(x, \cdot)$ equals L.

This notion of universal language was investigated in [NIR03] (under the name *weak universal language*). One of the results proved in that paper is that in most interesting cases, one could relax the condition "$\mathcal{U}(x, \cdot)$ equals L" to "$\mathcal{U}(x, \cdot)$ is a finite variant of L" without essentially changing the notion of universal language. In more detail:

Proposition 4.2. *[NIR03] Let C be different from the trivial classes $\{\emptyset\}$ and $\{\{0,1\}^*\}$. If a language $\mathcal{V}$ has the property that any language in C is a finite variant of $\mathcal{V}(x, \cdot)$ for some x, then then there is a language $\mathcal{U}$, universal for C, which is linear-time many-one reducible to $\mathcal{V}$.*

Thanks to this result, it is actually quite easy to characterize the complexity classes for which there are universal languages in one of the usual classes such as P, NP etc. By way of example:

Proposition 4.3. *For any m and any complexity class C, Σ_m^p contains a universal language for C iff there is k such that $\mathsf{C} \subseteq \Sigma_m\mathsf{TIME}(n^k)$. Analogously for Π_m^p and $\Pi_m\mathsf{TIME}$.*

Proof. Both directions are essentially obvious. For the "only if" part, we just use the fact that $\mathcal{U}(x, \cdot)$ cannot be more complex than $\mathcal{U}$ itself. The "if" part follows from proposition 4.2 and the proof of the (deterministic) time hierarchy theorem (cf. the proof of our theorem 3.5). □

An analogous characterization can be proved for the Δ levels of the polynomial hierarchy. Recall that Q_mSAT is the standard complete problem for Σ^p_m, so that $\Delta^p_m = \mathsf{P}^{Q_{m-1}SAT}$ (where we take Q_0SAT to be, say, the empty language).

Proposition 4.4. *For any m and any complexity class C, Δ^p_m contains a universal language for C iff there is k such that $\mathsf{C} \subseteq \mathsf{DTIME}^{Q_mSAT}(n^k)$.*

Two consequences of propositions 4.3-4.4 are apparent. Firstly, at least the levels of PH – but as a matter of fact, all the usual complexity classes – do not have universal languages for themselves.

Secondly, C has a universal language for C' exactly when C' is contained in an *initial segment* of C (in the sense of "initial segment" used in [ST95], where "an initial segment of P" means "a class of the form $\mathsf{DTIME}(n^k)$ for some k") – again, at least if C is a level of PH, but in fact it is a much more general phenomenon. Together with theorems 3.6 and 3.10, this suggests that despite some differences in the formulation, a universal language for C is essentially the same concept as an FM-truth definition for a logic $\mathcal{L}$ capturing C. In particular, define the vocabulary σ_1 to be $(\sigma_0 + P^1)$, where P is new (so that σ_1-models are in natural 1-1 correspondence with words over $\{0,1\}$). The following simple corollary, which formalizes some of this "essential identity of concepts", will play an important role in our further considerations:

Corollary 4.5. *Let C be a complexity class and let the logic $\mathcal{L}$ capture C over σ_1. Then for any m: there is a Σ^p_m universal language for C iff $\mathcal{L} \ll_{\sigma_1} \Sigma^1_m$.*

4.2 $\mathsf{P}^{\mathsf{NP}[n^r]}$ versus Σ^p_m and Δ^p_m

This and the following section form the core of the current chapter. They contain statements and proofs of the aforementioned results that if there are universal languages for NP within PH, they can be modified to yield universal languages for $\mathsf{P}^{\mathsf{NP}[n^r]}$, also within PH.

Theorem 4.6. *Let $m \geqslant 2$. Then:*

(A) If Δ^p_m contains a universal language for NP, then for every r, Δ^p_m contains a universal language for $\mathsf{P}^{\mathsf{NP}[n^r]}$.

(B) If both Σ^p_m and Π^p_m contain a universal language for NP, then for every r, both Σ^p_m and Π^p_m contain a universal language for $\mathsf{P}^{\mathsf{NP}[n^r]}$.

To prove the above two-part theorem, we will exploit yet another version of the Stewart normal form (cf. theorem 2.8), this time for second order sentences expressing $\mathsf{P}^{\mathsf{NP}[n^r]}$ properties:

Lemma 4.7. *Let σ be a vocabulary and let $r \geqslant 1$. Then any $\mathsf{P}^{\mathsf{NP}[n^r]}$ class of σ-models can be defined by an SO sentence of the form*

$$\exists R^r \, (\varphi(R) \,\&\, \neg\psi(R)),$$

where φ, ψ are Σ^1_1 formulae of vocabulary σ with R as the unique free variable.

The class of SO sentences in the form given by the lemma will be denoted by SNF_r. The proof of the lemma follows the one given by Gottlob in [Got97] very closely. Still, we present it here in detail, as some features of the construction will be needed in this and the following section.

Proof. Let $\mathcal{K}$ be a $\mathsf{P}^{\mathsf{NP}[n^r]}$ class of σ-models. Thus, there is a polynomial-time deterministic oracle machine T and an NP language $L \in \{0,1\}^*$ such that $\mathcal{K} = \{\mathbf{M} : \mathbf{M} \text{ is a } \sigma\text{-model and } T \text{ accepts } \mathbf{M} \text{ using } L \text{ as its oracle}\}$. Furthermore, T makes at most M^r oracle queries on input $\mathbf{M}$. We may assume w.l.o.g. that the number of queries is always exactly M^r. Thus, the string oans($\mathbf{M}$) of all oracle answers in the computation of T on input $\mathbf{M}$ (ordered chronologically) has length M^r. Recall the canonical correspondence between binary strings of length M^r and r-ary relations over M.

We introduce two classes of $(\sigma + R^r)$-models (where R is a new relational symbol). The classes $\mathcal{K}_1$ and $\mathcal{K}_2$ are defined as follows:

(1) Given $(\mathbf{M}, R)$, consider the machine T^+_R which behaves as T except that before making its i-th oracle query (for $i = 1, \ldots, M^r$), it looks at the i-th bit of the string corresponding to R – and then makes a regular query to L if the bit is 1, simulates a negative answer without querying if the bit is 0. $(\mathbf{M}, R)$ is in $\mathcal{K}_1$ iff: (a) all queries made by T^+_R on input $\mathbf{M}$ are answered positively, and (b) T^+_R accepts $\mathbf{M}$.

(2) Given $(\mathbf{M}, R)$, consider the machine T^-_R which behaves as T except that before making its i-th oracle query (for $i = 1, \ldots, M^r$), it looks at the i-th bit of the string corresponding to R – and then makes a query to L if the bit is 0, simulates a positive answer without querying if the bit is 1. $(\mathbf{M}, R)$ is in $\mathcal{K}_2$ iff all queries made by T^-_R on input $\mathbf{M}$ are answered negatively.

Note that $\mathcal{K}_1$ is in NP, as it is polynomial-time reducible to the language $(L\#)^*$, which is in NP by closure under conjunctions. Indeed, on input $(\mathbf{M}, R)$

the machine performing the reduction acts as T_R^+ on input $\mathbf{M}$, but instead of actually making an oracle query which T_R^+ makes (i.e. one where the appropriate bit of the string corresponding to R is 1), it simulates a positive answer, and writes down the queried string on the output tape, followed by a # symbol (strictly speaking, it doubles each bit of the queried string and writes 01 for #; see the *Preliminaries*). Additionally, if the computation does not end in an accepting state, the reducing machine attaches an arbitrary fixed no-instance of L to the list on the output tape, again followed by #. Clearly, it follows from the definition of $\mathcal{K}_1$ that $(\mathbf{M}, R) \in \mathcal{K}_1$ iff all the strings listed on the output tape are in L, or in other words, iff the word written on the output tape is in $(L\#)^*$.

A similar argument shows that $\mathcal{K}_2$ is in co-NP. Thus, let φ, ψ be Σ_1^1 sentences of vocabulary $(\sigma + R)$ such that $\mathcal{K}_1 = MOD(\varphi)$, $\mathcal{K}_2 = MOD(\neg\psi)$.

Consider now the second order sentence $\gamma := \exists R^r\,(\varphi(R)\,\&\,\neg\psi(R))$ (where $\varphi(R)$ is φ, now treated as a formula of vocabulary σ with R as a free variable; similarly for $\psi(R)$). Certainly, γ has the required form, so it remains to check that it defines $\mathcal{K}$.

Let $\mathbf{M} \in \mathcal{K}$ and let $R_0 \subseteq M^r$ be the relation corresponding to oans($\mathbf{M}$). Then $(\mathbf{M}, R_0)$ is in $\mathcal{K}_1$ and $\mathcal{K}_2$, so $\mathbf{M} \models \gamma$.

Let $\mathbf{M} \models \gamma$ and let $R \subseteq M^r$ be such that $(\mathbf{M}, R) \models \varphi$, $(\mathbf{M}, R) \models \neg\psi$. Then $(\mathbf{M}, R)$ is in both $\mathcal{K}_1$ and $\mathcal{K}_2$, so, by an inductive argument, the string corresponding to R is simply oans($\mathbf{M}$). This means that T_R^+ works on input $\mathbf{M}$ exactly as T does, and since $(\mathbf{M}, R) \in \mathcal{K}_1$, we know that T_R^+ accepts $\mathbf{M}$. Then so does T, and therefore $\mathbf{M}$ is in $\mathcal{K}$

Thus, the proof that γ defines $\mathcal{K}$ is completed. □

4.2.1 Versus Σ_m^p

Once we have the lemma, proving part (B) of theorem 4.6 presents no further difficulty. Fix m and assume that the hypothesis of (B) holds for m. By a slight generalization of corollary 4.5, it follows from this assumption that for any vocabulary σ, $\Sigma_1^1 \ll_\sigma \Sigma_m^1$, $\Sigma_1^1 \ll_\sigma \Pi_m^1$.

Fix r and let σ be σ_1, as defined at the end of the previous section. Let $Sat_\Sigma(x)$ (*resp.* $Sat_\Pi(x)$) be a Σ_m^1 (*resp.* Π_m^1) FM-truth definition for Σ_1^1 over the vocabulary $(\sigma_1 + R^r)$. Consider the following formula $Tr(x)$:

$$\exists \ulcorner\varphi\urcorner\, \exists \ulcorner\psi\urcorner\, (x = \ulcorner \exists R^r(\varphi(R)\,\&\,\neg\psi(R))\urcorner\,\&\,\exists R^r(Sat_\Sigma(\ulcorner\varphi\urcorner)\,\&\,\neg Sat_\Pi(\ulcorner\psi\urcorner))).$$

$Tr(x)$ is clearly equivalent to a Σ^1_m formula, and one may easily verify that it is an FM-truth definition for SNF_r over σ_1. By invoking corollary 4.5 again, we get a Σ^p_m universal language for $\mathsf{P}^{\mathsf{NP}[n^r]}$. The existence of the Π^p_m universal language follows from the closure of $\mathsf{P}^{\mathsf{NP}[n^r]}$ under complementation.

4.2.2 Versus Δ^p_m

To get part (A) of the theorem, we look again at the proof of lemma 4.7. This time, however, it will be convenient to diverge temporarily from our finite-model theoretical framework and adopt more standard complexity-theoretical terminology. Thus, models $\mathbf{M}$ become binary words x, r-ary relations R over $\mathbf{M}$ become $|x|^r$-long binary words y (so $(\mathbf{M}, R)$) becomes $\langle x, y\rangle$). The classes of models $\mathcal{K}$, $\mathcal{K}_1$ and $\mathcal{K}_2$ defined in the proof become languages over $\{0, 1\}$.

Note that the languages $\mathcal{K}_1$ and $\mathcal{K}_2$ have the following property: for any x and y, $|y| = |x|^r$, the pair $\langle x, y\rangle$ may only be in both $\mathcal{K}_i$'s if $y = \mathrm{oans}(x)$. So, we have:

$$x \in \mathcal{K} \text{ iff } \langle x, \mathrm{oans}(x)\rangle \in \mathcal{K}_1 \cap \mathcal{K}_2.$$

It follows that if we could somehow use a Δ^p_m universal language for NP to recover $\mathrm{oans}(x)$ given x, we could also use it to get a Δ^p_m universal language for $\mathsf{P}^{\mathsf{NP}[n^r]}$. To do this, we need just one more simple trick. Modify the definition of $\mathcal{K}_i$, for $i = 1, 2$, in the following way: $\langle x, y\rangle \in \tilde{\mathcal{K}}_i$ iff $|y|$ is *at most* $|x|^r$, and the answers to the queries of the machine T^+_y (T^-_y, respectively) are as required in the definition of $\mathcal{K}_i$ *as long as there are still bits of* y *to determine what the machine should do.* In other words, we do not care what happens with the i-th query of $T^{\pm}_y$ for $i > |y|$ (we may assume, e.g., that they are all given a NO answer without querying); in particular, if $|y| < |x|^r$, in the definition of $\tilde{\mathcal{K}}_1$ we do not worry about whether T^+_y accepts.

It is easy to see that $\tilde{\mathcal{K}}_1$ is still in NP and $\tilde{\mathcal{K}}_2$ is still in $\mathsf{co\text{-}NP}$. Furthermore, as in the proof of lemma 4.7, an inductive argument shows that for $|y| < |x|^r$, $\langle x, y\rangle$ is in $\tilde{\mathcal{K}}_1 \cap \tilde{\mathcal{K}}_2$ iff y is an initial subword of $\mathrm{oans}(x)$ (for $|y| = |x|^r$, there is the additional requirement that T accepts x). But this means that given oracles for $\tilde{\mathcal{K}}_1$ and $\tilde{\mathcal{K}}_2$, we can find $\mathrm{oans}(x)$ and determine whether $x \in \mathcal{K}$ by a polynomial-time prefix search.

Now fix m and assume that there is a Δ^p_m universal language $\mathcal{U}$ for NP. As a universal language, $\mathcal{U}$ supplies us with oracles for the $\tilde{\mathcal{K}}_i$'s! We thus

claim that a Δ_m^p universal language $\mathcal{U}_r$ for $\mathsf{P}^{\mathsf{NP}[n^r]}$ is given by the following algorithm:

$INPUT\ \langle\langle k_1, k_2\rangle, x\rangle$;
{we may assume that all inputs not of this form are rejected}
$y := \varepsilon$;
while $|y| < |x|^r$ **do**
    $y' := y0$;
    if $\neg\mathcal{U}(k_1, \langle x, y'\rangle)$ or $\mathcal{U}(k_2, \langle x, y'\rangle)$ **then** $y' := y1$;
    $y := y'$;
endwhile;
if $\mathcal{U}(k_1, \langle x, y\rangle)$ and $\neg\mathcal{U}(k_2, \langle x, y\rangle)$ **then** $ACCEPT$ **else** $REJECT$.

Clearly, $\mathcal{U}_r$ is a Δ_m^p language: the *while* loop is executed $|x|^r$ times, and it consists of two runs of the Δ_m^p algorithm for $\mathcal{U}$.

But it is also clear that $\mathcal{U}_r$ is universal for $\mathsf{P}^{\mathsf{NP}[n^r]}$. For if $\mathcal{K}$ is a $\mathsf{P}^{\mathsf{NP}[n^r]}$ language, and $\tilde{\mathcal{K}}_1$, $\tilde{\mathcal{K}}_2$ are as above, there will be indices k_1, k_2 such that $\mathcal{U}(k_1, \cdot)$ is $\tilde{\mathcal{K}}_1$ and $\mathcal{U}(k_2, \cdot)$ is the complement of $\tilde{\mathcal{K}}_2$. By the discussion above, it follows that after each execution of the *while* loop on input $\langle\langle k_1, k_2\rangle, x\rangle$, if $|y|$ is still smaller than $|x|^r$, then y is the unique word of length $|y|$ satisfying $\langle x, y\rangle \in \tilde{\mathcal{K}}_1 \cap \tilde{\mathcal{K}}_2$ – i.e., the unique $|y|$-bit long initial subword of $\mathrm{oans}(x)$. So, at the end of the loop, the algorithm for $\mathcal{U}_r$ finds $\mathrm{oans}(x)$. It then tests $\langle x, \mathrm{oans}(x)\rangle$ for membership in the $\tilde{\mathcal{K}}_i$'s. As already pointed out, this amounts to checking whether $x \in \mathcal{K}$.

In summary, $\mathcal{U}_r(\langle k_1, k_2\rangle, \cdot)$ is equal to $\mathcal{K}$. Since $\mathcal{K} \in \mathsf{P}^{\mathsf{NP}[n^r]}$ was arbitrary, $\mathcal{U}_r$ is indeed a Δ_m^p universal language for $\mathsf{P}^{\mathsf{NP}[n^r]}$ and part (A) of theorem 4.6 is proved.

As a side result, we get the following interesting corollary:

Corollary 4.8. *If* P^{NP} *has a universal language for* NP*, then the* $\{\mathsf{P}^{\mathsf{NP}[n^r]}\}_{r\in\omega}$ *hierarchy is strict and intertwines with the* P^{NP} *time hierarchy. That is, for any* r *there exists* k_r *such that all* $\mathsf{P}^{\mathsf{NP}[n^r]}$ *problems can be solved in* $\mathsf{DTIME}(n^{k_r})$ *with oracle* SAT*.*

Proof. If $\mathsf{P}^{\mathsf{NP}} = \Delta_2^p$ has a universal language for NP, then, by the theorem, it also has universal languages for all the $\mathsf{P}^{\mathsf{NP}[n^r]}$ classes. Thus, by proposition 4.4, for any given r there exists k_r such that all $\mathsf{P}^{\mathsf{NP}[n^r]}$ problems can be solved in $\mathsf{DTIME}(n^{k_r})$ with oracle SAT (actually, examination of the prefix search algorithm given above reveals that $k_r = O(r)$, where the constant implicit in

the $O(\cdot)$ notation depends only on the complexity of the universal language for NP).

Thus, the time hierarchy theorem relativized to SAT implies that $\mathsf{P}^{\mathsf{NP}[n^r]}$ is properly contained in P^{NP}, so the $\{\mathsf{P}^{\mathsf{NP}[n^r]}\}_{r\in\omega}$ hierarchy is infinite. Strictness follows using standard methods: if $\mathsf{P}^{\mathsf{NP}[n^r]} = \mathsf{P}^{\mathsf{NP}[n^{r+1}]}$, we use padding to get $\mathsf{P}^{\mathsf{NP}[n^{r+1}]} = \mathsf{P}^{\mathsf{NP}[n^{r+2}]}$, and so on. □

4.3 $\mathsf{P}^{\mathsf{NP}[n^r]}$ versus $\Sigma^p_m \cap \Pi^p_m$

Part (B) of theorem 4.6, with the awkward "both Σ^p_m and Π^p_m" requirements, is somewhat unsatisfactory. Could we replace "both Σ^p_m and Π^p_m" by just Σ^p_m? Evidently, there is a problem: Σ^1_m is not necessarily closed under complementation, so even if $Sat(x)$ is a Σ^1_m FM-truth definition for Σ^1_1,

$$\exists\ulcorner\varphi\urcorner\,\exists\ulcorner\psi\urcorner\,(x = \ulcorner\exists R^r(\varphi(R)\ \&\ \neg\psi(R))\urcorner\ \&\ \exists R^r(Sat(\ulcorner\varphi\urcorner)\ \&\ \neg Sat(\ulcorner\psi\urcorner)))$$

might not be equivalent to a Σ^1_m formula.

What about $\Sigma^p_m \cap \Pi^p_m$? Does a $\Sigma^p_m \cap \Pi^p_m$ universal language for NP mean a $\Sigma^p_m \cap \Pi^p_m$ universal language for $\mathsf{P}^{\mathsf{NP}[n^r]}$? This is not certain either: the previous section will guarantee us both a Σ^p_m and a Π^p_m universal language for $\mathsf{P}^{\mathsf{NP}[n^r]}$, but the proof given there cannot guarantee that these languages will be equal.

In terms of FM-truth definitions, the problem is roughly as follows. The Σ^1_m "FM-truth definition for $\mathsf{P}^{\mathsf{NP}[n^r]}$" determines whether a model satisfies a $\mathsf{P}^{\mathsf{NP}[n^r]}$ property $\mathcal{K}$ by using the Σ^1_m and Π^1_m FM-truth definitions for Σ^1_1 to evaluate the two Σ^1_1 components, φ and ψ, of the SNF_r sentence defining $\mathcal{K}$. What the Π^1_m "FM-truth definition for $\mathsf{P}^{\mathsf{NP}[n^r]}$" does is, more or less, to check whether the model is in *the complement* of $\mathcal{K}$ (and then negate the answer). This is carried out by applying the FM-truth definitions for Σ^1_1 to the components, φ' and ψ', of the SNF_r sentence defining this complement. But φ' and ψ' are entirely different from φ and ψ, so even if the two FM-truth definitions for Σ^1_1 are equivalent, they may be much better at evaluating one of those pairs of Σ^1_1 formulae than the other. The result may be that the Σ^1_m and Π^1_m "FM-truth definitions for $\mathsf{P}^{\mathsf{NP}[n^r]}$" will disagree as to whether $\mathcal{K}$ holds for some small models. Since there are infinitely many $\mathsf{P}^{\mathsf{NP}[n^r]}$ properties $\mathcal{K}$, and the notion of "small model" may vary wildly depending on $\mathcal{K}$, there is no clear way to make the Σ^1_m and Π^1_m "FM-truth definitions for $\mathsf{P}^{\mathsf{NP}[n^r]}$" equivalent.

In terms of universal languages, this phenomenon simply translates into the fact that that the word l_Σ indexing a given $\mathsf{P}^{\mathsf{NP}[n^r]}$ language L in the Σ^p_m universal language might be different from the word l_Π indexing L in the Π^p_m universal language, with no apparent way to get $l_\Sigma = l_\Pi$.

It is transparent from the above discussion that if we wanted to get a $\Sigma^p_m \cap \Pi^p_m$ universal language for $\mathsf{P}^{\mathsf{NP}[n^r]}$, it would help to have an additional assumption giving us some control over the behaviour of the FM-truth definitions for Σ^1_1 in small models. It turns out that there is a reasonably natural additional assumption of this kind, and that it is enough to get the $\Sigma^p_m \cap \Pi^p_m$ universal language, thus separating $\mathsf{P}^{\mathsf{NP}[n^r]}$ from $\Sigma^p_m \cap \Pi^p_m$:

Theorem 4.9. *If* $\mathsf{NTIME}(f) \subseteq \Sigma^p_m \cap \Pi^p_m$ *for some superpolynomial time-constructible function* f*, then for any* r *there is a* $\Sigma^p_m \cap \Pi^p_m$ *universal language for* $\mathsf{P}^{\mathsf{NP}[n^r]}$ *(and therefore* $\mathsf{P}^{\mathsf{NP}[n^r]} \subsetneq \Sigma^p_m \cap \Pi^p_m$*).*

The idea behind the proof is similar to the one used for theorem 4.6, but this time, some additional trickery is needed. We introduce a logic $\mathfrak{L}_r$ which captures $\mathsf{P}^{\mathsf{NP}[n^r]}$, and show that if $\Sigma^p_m \cap \Pi^p_m$ contains $\mathsf{NTIME}(f)$ for a superpolynomial time-constructible f, then for every σ, $\mathfrak{L}_r \ll_\sigma \Delta^1_m$ (i.e. there is a Σ^1_m FM-truth definition for $\mathfrak{L}_r$ equivalent to a Π^1_m formula).

Remark. Δ^1_m (formally defined as, say, the set of those Σ^1_m sentences which are equivalent to a Π^1_m sentence[2]) is not necessarily a logic, even under the liberal notion used in the thesis, so it is a slight abuse of notation to write $\mathfrak{L}_r \ll_\sigma \Delta^1_m$. Nevertheless, it remains true that if there is a Δ^1_m FM-truth definition for some $\mathcal{L}$ over some vocabulary σ, then there is a $\Sigma^p_m \cap \Pi^p_m$ property which is not $\mathcal{L}$-definable.

The logic $\mathfrak{L}_r$ we will use is defined as follows. For any vocabulary σ, the $\mathfrak{L}_r$-sentences over σ are ordered pairs $\langle T_1, T_2 \rangle$, where:

- T_1 is a (code of a) deterministic oracle machine, equipped with a polynomial time clock and a query counter which prohibits T_1 from asking more than n^r oracle queries on inputs of size n;
- T_2 is a (code of a) nondeterministic Turing machine equipped with a polynomial time clock.

[2]Of course, the equivalence is only required to hold over the class of models we consider, i.e. finite models with built-in arithmetic. Recall the well-known fact that if a Σ^1_1 sentence is equivalent to a Π^1_1 sentence in *all* (finite or infinite) models, then it is already equivalent to a first order sentence.

The semantics is straightforward: $\mathbf{M} \models \langle T_1, T_2 \rangle$ iff T_1 accepts $\mathbf{M}$ when using the language recognized by T_2 as its oracle.

Obviously, $\mathfrak{L}_r$ captures exactly $\mathsf{P}^{\mathsf{NP}[n^r]}$. As discussed, already if the hypothesis of theorem 4.6 part (B) holds for a given m, we have $\mathfrak{L}_r \ll_\sigma \Sigma^1_m$, $\mathfrak{L}_r \ll_\sigma \Pi^1_m$ for every σ. To obtain $\mathfrak{L}_r \ll_\sigma \Delta^1_m$, however, we will need the (apparently) stronger assumption that $\Sigma^p_m \cap \Pi^p_m$ contains a superpolynomial nondeterministic time class.

We will also need a simple but important observation on FM-representing computable relations (or, more generally, recursively enumerable relations).

Definition 4.10. Given a relation $R \subseteq \omega^n$ and a formula $\varphi_R(\mathbf{x})$ which FM-represents it, call φ_R *decent* if it has the property that for any $\mathbf{a} \in \omega^n$, if $\mathbf{M} \models \varphi_R(\mathbf{a})$ for some model $\mathbf{M}$, then also $\mathbf{M}' \models \varphi_R(\mathbf{a})$ whenever $M' > M$.

Remark. It follows from definition 4.10 and from the definition of FM-representation that given a relation R FM-represented by a decent formula $\varphi_R(\mathbf{x})$ and any tuple $\mathbf{a}$, if there exists a model $\mathbf{M}$ such that $\mathbf{M} \models \varphi_R(\mathbf{a})$, then $\mathbf{a} \in R$.

Thus, a decent formula is one which "never falsely claims" that some tuple is in the relation it FM-represents, and additionally "never withdraws such a claim" when passing to larger models. The observation is:

Proposition 4.11. *For any recursively enumerable relation $R \subseteq \omega^n$, there is a decent formula which FM-represents it.*

Proof. This fact is implicitly contained in the proof of the FM-representability theorem in [Mos01]; we give an explicit argument nonetheless. If $R \subseteq \omega^n$ is RE, then it is defined in $\mathbb{N}$ by a Σ^0_1 formula, i.e. an FO-formula of the form $\exists \mathbf{y} \psi(\mathbf{x}, \mathbf{y})$ where ψ consists of a string of *bounded* quantifiers followed by a quantifier-free matrix. It is well-known that we may assume w.l.o.g. that the quantifiers in ψ are bounded by variables and not by complex terms (see e.g. [HP93]).

The matrix of ψ is a boolean combination of equalities between some polynomials in $\mathbf{x}, \mathbf{y}$, and $\mathbf{z}$, where $\mathbf{z}$ is the tuple of variables which are quantified in ψ. Therefore, since the z's are bounded from above by the x's and/or y's, there is a polynomial $p(\mathbf{x}, \mathbf{y})$ such that for any choice of tuples $\mathbf{a}, \mathbf{b}$ to interpret $\mathbf{x}, \mathbf{y}$ respectively, the truth of $\psi(\mathbf{a}, \mathbf{b})$ does not depend on any number greater than $p(\mathbf{a}, \mathbf{b})$.

Let $\varphi_R(\mathbf{x})$ be $\exists \mathbf{y} \xi(\mathbf{x}, \mathbf{y})$, where ξ is ψ modified in the following way: the original quantifier prefix of ψ is left unchanged, and the matrix is replaced

by the conjunction of an FO-formula expressing "there exists $p(\mathbf{x}, \mathbf{y})$" and a formula which arises from the original matrix by eliminating all the complex terms (i.e. substituting $\exists w(+(x_1, x_2, w) \,\&\, w = x_3)$ for $x_1 + x_2 = x_3$ etc.). It is not difficult to see that φ_R is a decent formula which FM-represents R. $\square$

Actually, one can show that being FM-represented by some decent formula is exactly equivalent to being RE. We leave out the (easy) proof of this fact as we do not need it.

We now prove the crucial lemma of this section. It states that if the hypothesis of theorem 4.9 holds for m, then there exist Σ^1_m and Π^1_m FM-truth definitions for Σ^1_1 which are, in a sense, well-behaved.

Lemma 4.12. *If $\Sigma^p_m \cap \Pi^p_m$ contains $\mathsf{NTIME}(f)$ for some superpolynomial time-constructible f, then for any vocabulary σ there exists a Σ^1_m formula $Sat^+_\Sigma(x)$ and an equivalent Π^1_m formula $Sat^+_\Pi(x)$ such that:*

- *$Sat^+_\Sigma(x)$ and $Sat^+_\Pi(x)$ are FM-truth definitions for Σ^1_1 over σ;*
- *there is a computable function which assigns to a Σ^1_1 sentence ψ over σ a number $\mathrm{bnd}(\psi)$ such that Sat^+_Σ and Sat^+_Π recognize the truth value of ψ correctly in all σ-models of cardinality greater than $\mathrm{bnd}(\psi)$.*

Proof. Fix σ. From now on, any model $\mathbf{M}$ appearing in the proof will be a σ-model.

We may assume w.l.o.g. that $f(n)$ is of the form $n^{g(n)}$, where g is some computable function satisfying $lim_{n\to\infty} g(n) = \infty$; if not, then we may find such g for which $n^{g(n)} \leqslant f(n)$ — where it is not required for $n^{g(n)}$ to be time-constructible, computability will suffice.

Clearly, there exists a computable function h such that $g(h(n)) \geqslant n$ for all n.

It is well-known that there exists a Turing machine T which solves the problem:

"given input $(\mathbf{M}, \ulcorner\psi\urcorner)$, where ψ is a Σ^1_1 sentence,
is it true that $\mathbf{M} \models \psi$?"

in nondeterministic time $M^{\mathrm{lh}(\psi)}$ (see e.g. [GLV99]). So, the machine $\tilde{T}$ which, on input $(\mathbf{M}, \ulcorner\psi^\frown 0^i\urcorner)$, disregards the string 0^i and simulates T on $(\mathbf{M}, \ulcorner\psi\urcorner)$ actually works in $\mathsf{NTIME}(M^{g(\mathrm{lh}(\psi^\frown 0^i))})$ — and hence in $\mathsf{NTIME}(f)$ — when restricted to inputs of the form $(\mathbf{M}, \ulcorner\psi^\frown 0^{h(\mathrm{lh}(\psi))-\mathrm{lh}(\psi)}\urcorner)$.

Since the function h is computable, we know by proposition 4.11 that there is a decent formula $\varphi_h(x, y)$ which FM-represents its graph. Consider now a machine T^* which on input $(\mathbf{M}, \ulcorner \psi^\frown 0^i \urcorner)$ does the following:

- using the time-constructibility of f, keep a clock for, say, $2f$; if a computation tries to use more than this allotted amount of time, stop and reject;
- check whether $\mathbf{M} \models \varphi_h(\mathrm{lh}(\psi), i + \mathrm{lh}(\psi))$; if not, reject;
- else work as $\tilde{T}$ on the input and accept iff $\tilde{T}$ does.

Clearly, for all sufficiently large $\mathbf{M}$, the machine T^* accepts $(\mathbf{M}, \ulcorner \psi^\frown 0^i \urcorner)$ if and only if $i = h(\mathrm{lh}(\psi)) - \mathrm{lh}(\psi)$ and $\mathbf{M} \models \psi$. Moreover, T^* works in $\mathsf{NTIME}(O(f)) \subseteq \mathsf{NTIME}(f) \subseteq \Sigma^p_m \cap \Pi^p_m$, so there is a Σ^1_m formula $\varphi^*_\Sigma(x)$ — equivalent to a Π^1_m formula $\varphi^*_\Pi(x)$ — which is true of w in $\mathbf{M}$ iff T^* accepts $(\mathbf{M}, w)$.

Let $Sat^+_\Sigma(x)$ be

$$\exists w \,(\text{“}w = \ulcorner x^\frown z \urcorner \text{ where } z \text{ is a string of zeroes”} \\ \& \, \varphi_h(\mathrm{lh}(x), \mathrm{lh}(w)) \,\&\, \varphi^*_\Sigma(w)),$$

and let $Sat^+_\Pi(x))$ be defined analogously using φ^*_Π. It is not hard to see that $Sat^+_\Sigma(x)$ and $Sat^+_\Pi(x)$ are equivalent FM-truth definitions for Σ^1_1 over σ. It therefore remains to check that given ψ, we can compute $\mathrm{bnd}(\psi)$ such that $Sat^+_\Sigma(x)$ works properly for ψ in all models larger than $\mathrm{bnd}(\psi)$.

Before we describe the algorithm, we note that since ψ_h is an FO-formula, testing $\mathbf{M} \models \varphi_h(\mathrm{lh}(\psi), i + \mathrm{lh}(\psi))$ on input $(\mathbf{M}, \ulcorner \psi^\frown 0^i \urcorner)$ requires (deterministic) time M^l for some fixed l independent of ψ. By assumption, f is superpolynomial, so there is an n_0 such that for all $n \geqslant n_0$, $f(n) \geqslant n^l$.

Now compute $\mathrm{bnd}(\psi)$ on input ψ as follows. Find the smallest model $\mathbf{M}'$ in which there is a w such that $\mathbf{M}' \models \chi(w)$, where $\chi(w)$ is

$$\text{“}w = \ulcorner \psi^\frown z \urcorner \text{ where } z \text{ is a string of zeroes”} \,\&\, \varphi_h(\mathrm{lh}(\psi), \mathrm{lh}(w)).$$

Since we may assume that both the conjuncts in χ are decent, this w will be equal to $\ulcorner \psi^\frown 0^{h(\mathrm{lh}(\psi)) - \mathrm{lh}(\psi)} \urcorner$ and will also satisfy χ in all models larger than $\mathbf{M}'$. Let $\mathrm{bnd}(\psi) := \max(M', n_0)$.

Given any $(\mathbf{M}, w)$ with $M \geqslant \mathrm{bnd}(\psi)$, the machine T^* will recognize that w is of the proper length in time $M^l \leqslant f(M)$, and check whether $\tilde{T}$ accepts

$(\mathbf{M}, w)$ in time $f(M)$. Thus, T^* will not exceed its time limit $2f(M)$, so it will correctly answer whether $\mathbf{M} \models \psi$, which proves that $\mathrm{bnd}(\psi)$ is indeed large enough. □

We need one more observation: two important constructions associated with the logic $\mathfrak{L}_r$ are effective and can therefore be FM-represented by decent formulae.

Lemma 4.13. *There exist computable functions:*

(a) neg*, which assigns to (the Gödel number of) an $\mathfrak{L}_r$-sentence ξ (the Gödel number of) the "negation" of ψ, i.e. an $\mathfrak{L}_r$-sentence* $\mathrm{neg}(\xi)$ *which is true in exactly those models in which ξ is false;*

(b) snf*, which assigns to (the Gödel number of) an $\mathfrak{L}_r$-sentence ξ (the Gödel number of) a sentence* $\mathrm{snf}(\xi)$ *in SNF_r such that ξ and* $\mathrm{snf}(\xi)$ *are equivalent.*

Proof. For (a), let $\xi = \langle T_1, T_2 \rangle$. Then $\mathrm{neg}(\ulcorner \xi \urcorner)$ is $\ulcorner \langle \overline{T_1}, T_2 \rangle \urcorner$, where $\overline{T_1}$ is T_1 with accepting and rejecting states interchanged.

For (b), we just need to make sure that the construction in the proof of lemma 2.8 can be carried out in an effective way. We sketch the argument. Given $\ulcorner \xi \urcorner = \ulcorner \langle T_1, T_2 \rangle \urcorner$, we may effectively construct NP-machines $T_{\mathcal{K}_1}$, $T_{\mathcal{K}_2}$ which recognize $\mathcal{K}_1$ and the complement of $\mathcal{K}_2$, respectively. Moreover, we can compute a time bound for both of these machines: if T_1 has an n^{k_1} clock and T_2 has an n^{k_2} clock, then $T_{\mathcal{K}_1}$ essentially runs as T_1 except that it will have to simulate T_2 at most n^r times (the exact number depends on the additional input relation R) on strings of length $\leqslant n^{k_1}$ (since T_1 will not have the time to write any longer strings on its oracle tape). So a rough estimate of the time needed by $T_{\mathcal{K}_1}$ is $n^{k_1} + n^{r+k_1 \cdot k_2}$. Some additional time is needed for looking at the bits of R, so we may take $n^{r+k_1 \cdot k_2+1}$ as a suitable upper bound. The same bound will do for $T_{\mathcal{K}_2}$.

Once $T_{\mathcal{K}_1}$, $T_{\mathcal{K}_2}$, and their time bounds are known, we can use them to compute the Σ^1_1 sentences φ, ψ which describe the action of $T_{\mathcal{K}_1}$ and $T_{\mathcal{K}_2}$, respectively. We then set $\gamma := \exists R^r \, (\varphi(R) \,\&\, \neg\psi(R))$ and $\mathrm{snf}(\ulcorner \xi \urcorner) := \ulcorner \gamma \urcorner$. □

By proposition 4.11, there exist decent formulae which represent the graphs of neg and snf. Choose some such formulae and call them $\mathrm{neg}(x_1, x_2)$ and $\mathrm{snf}(x_1, x_2)$, respectively.

We are now ready to complete the proof of theorem 4.9. Fix m and assume that the hypothesis of the theorem holds for m. Fix r and σ. Let $Sat^+_\Sigma(x)$ and $Sat^+_\Pi(x)$ be the equivalent Σ^1_m and Π^1_m FM-truth definitions for Σ^1_1 over $(\sigma + R^r)$ given by lemma 4.12. Let bnd be the function from that lemma appropriate for Sat^+_Σ and Sat^+_Π.

Recall that bnd is a computable function. Thus, by a minor modification of the proof of proposition 4.11, there is a first order σ_0-formula large(x) which satisfies the following three requirements for any given Σ^1_1 sentence ψ over $(\sigma + R^r)$:

- $\mathbf{M} \models \text{large}(\ulcorner\psi\urcorner)$ implies $M \geqslant \text{bnd}(\psi)$;
- $\mathbf{M} \models \text{large}(\ulcorner\psi\urcorner)$ for all sufficiently large $\mathbf{M}$;
- if $\mathbf{M} \models \text{large}(\ulcorner\psi\urcorner)$ and $M' \geqslant M$, then $\mathbf{M}' \models \text{large}(\ulcorner\psi\urcorner)$.

Consider now the formula good$(x, x', y, y', \ulcorner\varphi\urcorner, \ulcorner\psi\urcorner, \ulcorner\varphi'\urcorner, \ulcorner\psi'\urcorner)$:

$$\begin{gathered}\text{neg}(x, x') \,\&\, \text{snf}(x, y) \,\&\, \text{snf}(x', y') \\ \&\; y = \ulcorner\exists R^r(\varphi(R) \,\&\, \neg\psi(R))\urcorner \,\&\, y' = \ulcorner\exists R^r(\varphi'(R) \,\&\, \neg\psi'(R))\urcorner \\ \&\; \text{large}(\ulcorner\varphi\urcorner) \,\&\, \text{large}(\ulcorner\psi\urcorner) \,\&\, \text{large}(\ulcorner\varphi'\urcorner) \,\&\, \text{large}(\ulcorner\psi'\urcorner).\end{gathered}$$

Let $Tr_\Sigma(x)$ be:

$$\exists x' \,\ldots\, \exists\ulcorner\psi'\urcorner\,(\text{good}(x, x', \ldots, \ulcorner\psi'\urcorner) \,\&\, \exists R^r(Sat^+_\Sigma(\ulcorner\varphi\urcorner) \,\&\, \neg Sat^+_\Pi(\ulcorner\psi\urcorner))),$$

and let $Tr_\Pi(x)$ be:

$$\exists x \,\ldots\, \exists\ulcorner\psi'\urcorner\,(\text{good}(x, x', \ldots, \ulcorner\psi'\urcorner) \,\&\, \neg\exists R^r(Sat^+_\Sigma(\ulcorner\varphi'\urcorner) \,\&\, \neg Sat^+_\Pi(\ulcorner\psi'\urcorner))).$$

It is not hard to see that Tr_Σ is equivalent to a Σ^1_m formula, that Tr_Π is equivalent to a Π^1_1 formula, and that both Tr_Σ and Tr_Π are FM-truth definitions for $\mathfrak{L}_r$ over σ. Let us then check that Tr_Σ and Tr_Π are equivalent.

It may be assumed that neither $Tr_\Sigma(n)$ nor $Tr_\Pi(n)$ ever holds if n is not the Gödel number of an $\mathfrak{L}_r$-sentence. So let $\mathbf{M}$ be a σ-model and let $\ulcorner\xi\urcorner \in M$ for some $\mathfrak{L}_r$-sentence ξ. If there is no tuple $(x', \ldots, \ulcorner\psi'\urcorner)$ such that $\mathbf{M} \models \text{good}(\ulcorner\xi\urcorner, x', \ldots, \ulcorner\psi'\urcorner)$, then we have both $\mathbf{M} \not\models Tr_\Sigma(\ulcorner\xi\urcorner)$ and $\mathbf{M} \not\models Tr_\Pi(\ulcorner\xi\urcorner)$.

Otherwise, by the decency of the formulae neg(x_1, x_2) and snf(x_1, x_2), it must be the case that $x' = \text{neg}(\ulcorner\xi\urcorner)$, that $y = \text{snf}(\ulcorner\xi\urcorner) = \ulcorner\exists R^r(\varphi(R) \,\&$

$\neg\psi(R))^\urcorner$, and that $y' = \mathrm{snf}(\mathrm{neg}(^\ulcorner\xi^\urcorner)) = {}^\ulcorner\exists R^r(\varphi'(R)\ \&\ \neg\psi'(R))^\urcorner$. Moreover, by the choice of the formula $\mathrm{large}(x)$, M is large enough for Sat^+_Σ and Sat^+_Π to correctly recognize, given any $R \subseteq M^r$, the truth value of φ, ψ, φ', and ψ' in $(\mathbf{M}, R)$. Thus, $\mathbf{M} \models Tr_\Sigma(^\ulcorner\xi^\urcorner)$ iff $\mathbf{M} \models \exists R^r\ (\varphi(R)\ \&\ \neg\psi(R))$, that is, iff $\mathbf{M} \models \xi$. Similarly, $\mathbf{M} \models Tr_\Pi(^\ulcorner\xi^\urcorner)$ iff $\mathbf{M} \not\models \exists R^r\ (\varphi'(R)\ \&\ \neg\psi'(R))$, that is, again, iff $\mathbf{M} \models \xi$. So, also in this case $\mathbf{M} \models Tr_\Sigma(^\ulcorner\xi^\urcorner)$ iff $\mathbf{M} \models Tr_\Pi(^\ulcorner\xi^\urcorner)$, which proves that Tr_Σ and Tr_Π are indeed equivalent.

Hence, for any choice of r and σ, we have found a Δ^1_m FM-truth definition for $\mathfrak{L}_r$ over σ. For the particular choice of $\sigma := \sigma_1$, the FM-truth definition for $\mathfrak{L}_r$ easily translates into a $\Sigma^p_m \cap \Pi^p_m$ universal language for $\mathsf{P}^{\mathsf{NP}[n^r]}$. This ends the proof of theorem 4.9.

4.4 Concluding remarks

I. There are no obstacles to extending the methods of sections 2 and 3 to $\mathsf{P}^{\Sigma^p_i[n^r]}$ instead of just $\mathsf{P}^{\mathsf{NP}[n^r]}$. Thus, for any i, we obtain the following result:

Theorem 4.14. *Let $m \geqslant i+1$. Then:*

(A) If there is a Δ^p_m universal language for Σ^p_i, then for every r, there also is a Δ^p_m universal language for $\mathsf{P}^{\Sigma^p_i[n^r]}$;

(B) If there are both a Σ^p_m and a Π^p_m universal language for Σ^p_i, then for every r, there are both a Σ^p_m and a Π^p_m universal language for $\mathsf{P}^{\Sigma^p_i[n^r]}$.

(C) If there exists a superpolynomial time-constructible function f such that $\Sigma_i\mathsf{TIME}(f) \subseteq \Sigma^p_m \cap \Pi^p_m$, then there is a $\Sigma^p_m \cap \Pi^p_m$ universal language for $\mathsf{P}^{\Sigma^p_i[n^r]}$ (and hence $\mathsf{P}^{\Sigma^p_i[n^r]} \subsetneq \Sigma^p_m \cap \Pi^p_m$).

II. Recall that L^{NP} is equal to $\mathsf{P}^{\mathsf{NP}[O(\log n)]}$, the class of languages recognizable in deterministic polynomial time with just logarithmically many queries to an NP oracle ([Wag90],[BH91]). Thus, for example, if there exists a Δ^p_m universal language for NP, then there also is a Δ^p_m universal language for L^{NP}. Obviously, appropriate analogues of part (B) of theorem 4.6, and of theorem 4.9, also hold.

III. Define $(\Sigma^1_1)^{\leqslant r}$ as the subclass of Σ^1_1 consisting of those formulae in which the existential second order quantifiers have arity at most r. The hierarchy

$\langle(\Sigma^1_1)^{\leqslant r}\rangle_{r\in\omega}$ is sometimes referred to as the Σ^1_1 arity hierarchy. This hierarchy is known to be strict if we take into account vocabularies of arbitrary arity ([Ajt83]), but its strictness over a uniform vocabulary remains an open problem.

Observe that if model checking for FO over any fixed vocabulary is contained in $\mathsf{NTIME}(n^k)$ — the number k may depend on σ — then, by corollary 3.6, there is a Σ^1_1 FM-truth definition for FO over any vocabulary σ. It is a routine task to transform these definitions into Σ^1_1 FM-truth definitions for $(\Sigma^1_1)^{\leqslant r}$, for any r and over any σ. We therefore have:

Proposition 4.15. *If for any σ there is k such that model checking for FO over σ is in $\mathsf{NTIME}(n^k)$, then for any σ and r there is k_r such that model checking for $(\Sigma^1_1)^{\leqslant r}$ over σ is in $\mathsf{NTIME}(n^{k_r})$ (and thus the Σ^1_1 arity hierarchy is infinite — hence, by [Fag75], strict — over any σ).*

Note that since model-checking for FO is contained in L, corollary 4.5 (and a slight extension thereof) implies that the hypothesis of the proposition is satisfied if there is an NP universal language for L — in particular, if NP contains $\mathsf{DSPACE}(f)$ for any space-constructible function f which dominates log.

Again, this can be generalized to arity hierarchies of the higher prenex classes of SO. Let $(\Sigma^1_m)^{\leqslant r}$ denote the subclass of Σ^1_m consisting of formulae in which the second order quantifiers in the initial existential quantifier block have arity at most r (the arity of the other relational quantifiers is arbitrary). We have:

Proposition 4.16. *If there is k such that $\Pi^p_{m-1} \subseteq \Sigma_m\mathsf{TIME}(n^k)$ (i.e., if Σ^p_m contains a universal language for Π^p_{m-1}), then for any σ and r there is k_r such that model checking for $(\Sigma^1_m)^{\leqslant r}$ over σ is in $\Sigma_m\mathsf{TIME}(n^{k_r})$.*

Note again that the hypothesis will hold e.g. if Σ^p_m contains $\Pi_{m-1}\mathsf{TIME}(f)$ for a time-constructible superpolynomial f.

IV. As pointed out at the beginning of the chapter, our theorems 4.6 and 4.9 have as its consequence the (already known) result that for any r, $\mathsf{P}^{\mathsf{NP}[n^r]}$ is properly contained in NEXP. It is perhaps worth noting that the largest syntactically defined fragments of SO which can be separated from NEXP by a similar argument are the closures of Σ^1_1 under second order quantifiers of bounded arity. For, let $SO^{\leqslant r}(\Sigma^1_1)$ denote the closure of Σ^1_1 under boolean

connectives, first order quantification, and second order quantification over relations of arity at most r. Then we easily get:

Proposition 4.17. *If there is a* PSPACE *universal language for* NP*, then for any σ and r there exists k_r such that all $SO^{\leqslant r}(\Sigma^1_1)$-definable classes of σ-models are in* $\mathsf{DSPACE}(n^{k_r})$*,*

from which it follows that $SO^{\leqslant r}(\Sigma^1_1)$ cannot define all NEXP classes of models.

V. We have concentrated on the relations between the existence of universal languages for various classes within the polynomial hierarchy. It is just as possible, however, to apply the techniques developed in this chapter to higher complexity classes. Consider, for example, the class $\mathsf{P}^{\mathsf{NEXP}}$, equal to $\mathsf{PSPACE}^{\mathsf{NP}}$ and to L. Hemaspaandra's strong exponential hierarchy (see section 2.3). We can consider the version of the Stewart normal form for $\mathsf{P}^{\mathsf{NEXP}}$ properties given by lemma 2.8, and proceed as in sections 4.2 and 4.3 to obtain the following analogue of theorems 4.6 and 4.9 (Δ^{exp}_m is a piece of notation for $\mathsf{EXP}^{\Sigma^p_{m-1}}$):

Theorem 4.18. *For any $m \geqslant 2$:*

(A) If there is a Δ^{exp}_m universal language for NEXP*, then there also is a Δ^{exp}_m universal language for $\mathsf{P}^{\mathsf{NEXP}}$;*

(B) If there are both a Σ^{exp}_m and a Π^{exp}_m universal language for NEXP*, then there are both a Σ^{exp}_m and a Π^{exp}_m universal language for $\mathsf{P}^{\mathsf{NEXP}}$.*

(C) If there exists a time-constructible function f which dominates 2^{n^k} for all k and satisfies $\mathsf{NTIME}(f) \subseteq \Sigma^{exp}_m \cap \Pi^{exp}_m$, then there is a $\Sigma^{exp}_m \cap \Pi^{exp}_m$ universal language for $\mathsf{P}^{\mathsf{NEXP}}$ (and hence $\mathsf{P}^{\mathsf{NEXP}} \subsetneq \Sigma^{exp}_m \cap \Pi^{exp}_m$).

As previously in the case of $\mathsf{P}^{\mathsf{NP}[n^r]}$, any of the three parts of theorem 4.18 can be used to get the following corollary:

Corollary 4.19. $\mathsf{P}^{\mathsf{NEXP}} \subsetneq \mathsf{2NEXP}$.

Appendix A

Proof of lemma 2.8

We proceed as in the proof of lemma 4.7, i.e. as in the proof of corollary 5.3 in [Got97] originally. The one change is that we have to take the effects of padding into account.

So let $\mathcal{K}$ be a class of σ-models in $(\mathsf{r}-2)\mathsf{EXP}^{(\mathsf{d}-\mathsf{r}+1)\Sigma_m\mathsf{EXP}}$. $\mathcal{K}$ is then recognized by a deterministic oracle machine T running in $(r-2)$-fold exponential time and using a language $L \in (\mathsf{d}-\mathsf{r}+1)\Sigma_m\mathsf{EXP}$ as oracle. Due to its time bound, T makes at most $\exp_{r-2}(M^k)$ oracle queries on input $\mathbf{M}$ (for some k). So oans($\mathbf{M}$) will have length at most $\exp_{r-2}(M^k)$. Binary strings of length $\exp_{r-2}(M^k)$ are in natural correspondence with relations of type

$$\tau = \overbrace{(\ldots(}^{r-1}\underbrace{\iota\ldots\iota}_{k}\overbrace{)\ldots)}^{r-1}$$

over M. Note that order(τ) $= r$.

As in lemma 4.7, we introduce two auxiliary classes of models, $\mathcal{K}_1$ and $\mathcal{K}_2$. This time, $\mathcal{K}_1$ and $\mathcal{K}_2$ are classes of σ–models expanded by an additional relation $\mathbf{R}^\tau$ of type τ. The definitions are fully analogous to the previous ones:

(1) Given $(\mathbf{M}, \mathbf{R}^\tau)$, R, consider the machine $T^+_{\mathbf{R}}$ which behaves as T except that before making its i-th oracle query, it looks at the i-th bit of the string corresponding to $\mathbf{R}$ — and then makes a regular query to L if the bit is 1, simulates a negative answer without querying if the bit is 0. $(\mathbf{M}, \mathbf{R})$ is in $\mathcal{K}_1$ iff: (a) all queries regularly made by $T^+_{\mathbf{R}}$ on input $\mathbf{M}$ are answered positively, and (b) $T^+_{\mathbf{R}}$ accepts $\mathbf{M}$.

(2) Given $(\mathbf{M}, \mathbf{R}^\tau)$, consider the machine $T^-_{\mathbf{R}}$ which behaves as T except that before making its i-th oracle query, it looks at the i-th bit of the string corresponding to R — and then makes a regular query to L if the bit is 0, simulates a positive answer without querying if the bit is 1. $(\mathbf{M}, \mathbf{R})$ is in $\mathcal{K}_2$ iff all queries regularly made by $T^-_{\mathbf{R}}$ on input $\mathbf{M}$ are answered negatively.

We now need to determine the complexity of the $\mathcal{K}_i$'s. Reasoning as in the proof of lemma 4.7, we use the knowledge that L is in $(\mathsf{d}-\mathsf{r}+1)\Sigma_m\mathsf{EXP}$ to get that $\mathcal{K}_1$ is also in $(\mathsf{d}-\mathsf{r}+1)\Sigma_m\mathsf{EXP}$ *with respect to the whole input* $(\mathbf{M}, \mathbf{R}^\tau)$. Similarly, $\mathcal{K}_2$ is in $(\mathsf{d}-\mathsf{r}+1)\Sigma_m\mathsf{EXP}$ with respect to the input $(\mathbf{M}, \mathbf{R}^\tau)$.

But this means that $\mathcal{K}_1$ needs $(d-1)$-fold exponential time on a Σ_m machine if we consider just $\mathbf{M}$ to be the input. $\mathcal{K}_2$ needs $(d-1)$-fold exponential time w.r.t. $\mathbf{M}$ on a Π_m machine. So, $\mathcal{K}_1$ and the complement of $\mathcal{K}_2$ are both defined by Σ^d_m formulae with $\mathbf{R}$ as a free variable. Call these Σ^d_m formulae $\varphi(\mathbf{R})$ and $\psi(\mathbf{R})$, respectively.

The rest of the proof contains nothing new: one considers the sentence $\exists\mathbf{R}(\varphi(\mathbf{R}) \,\&\, \neg\psi(\mathbf{R}))$ and checks that it defines $\mathcal{K}$ as required. □

Appendix B

An FM-truth definition

As we have seen in chapter 3 (in particular, in theorem 3.5), it is often not difficult at all to prove general results on which logics have FM-truth definitions for which. Writing down the actual definitions, however, usually requires additional work. We illustrate this by describing how to construct a Σ^2_2 FM-truth definition for $[\Sigma^2_2]^{\leqslant k}$, known to exist by corollary 3.9.

The exact choice of k and the vocabulary is immaterial, but an additional technical notion will be helpful:

Definition B.1. A *simple type* is a type of the form $\tau = \overbrace{(\ldots(}^{d}\underbrace{\iota\ldots\iota}_{k}\overbrace{)\ldots)}^{d}$ for some $d, k \in \omega$.

Let $s[\Sigma^d_m]^{\leqslant k}$ be $[\Sigma^d_m]^{\leqslant k}$ restricted to variables of simple types only. By an argument similar to the one in the proof of theorem 2.5, it can be shown that $s[\Sigma^d_m]^{\leqslant k}$ captures at least $\Sigma_m\mathsf{TIME}(\exp_{d-1}(n^k))$. In particular then, we always have $[\Sigma^d_m]^{\leqslant k} < s[\Sigma^d_m]^{\leqslant k+1}$. Therefore, it is enough to give a Σ^2_2 FM-truth definition for $s[\Sigma^2_2]^{\leqslant k}$ — the proof of theorem 3.4 then shows how to transform it into a Σ^2_2 FM-truth definition for $[\Sigma^2_2]^{\leqslant k-1}$.

Observe further that it is enough to write an FM-truth definition for the fragment of $s[\Sigma^2_2]^{\leqslant k}$ in which all second and third order variables have basic arity exactly k (there is an effective procedure which, given a $s[\Sigma^2_2]^{\leqslant k}$ sentence, outputs a logically equivalent $s[\Sigma^2_2]^{\leqslant k}$ sentence of the required form; the procedure consists in "adding superfluous arguments").

We treat first order valuations as unary functions on the universe. If $\ulcorner x \urcorner$ is the Gödel number of some first order variable x, and h is a unary function

on M (where $\mathbf{M} \models$ "$\ulcorner x \urcorner$ is a first order variable"[1]), then the valuation h assigns to x the value $h(\ulcorner x \urcorner)$. Formally, of course, a unary function is a binary relation satisfying the usual existence and uniqueness conditions.

Similarly, we treat second order valuations for k-ary relational variables as $(k+1)$-ary relations: if $\ulcorner R \urcorner$ is the Gödel number of some k-ary relational variable R, W is a $(k+1)$-ary relation on M (where $\mathbf{M} \models$ "$\ulcorner R \urcorner$ is a k-ary relational variable"), and $a_1, \ldots, a_k$ are elements of M, then the tuple $(a_1, \ldots, a_k)$ is in the relation assigned to R by W iff $W(\ulcorner R \urcorner, a_1, \ldots, a_k)$ holds.

Valuations for third order variables of type $((\underbrace{\iota \ldots \iota}_{k}))$ are treated in a similar way, although we have to quantify them differently so as to keep our truth definition within Σ^2_2. The FM-truth definition now reads (x is the free first order variable; third order variables are in boldface and of appropriate type; the phrases in quotation marks should be replaced by appropriate decent FM-representing formulae):

$$\exists x_1\, \exists x_2\, \exists x_3 \left\langle (x = x_1 * x_2 * x_3)$$

& "x_1 is a string of existentially quantified variables of type $((\underbrace{\iota \ldots \iota}_{k}))$"

& "x_2 is a string of universally quantified variables of type $((\underbrace{\iota \ldots \iota}_{k}))$"

& "x_3 is a second order formula with all relation variables of type $(\underbrace{\iota \ldots \iota}_{k})$ and possibly containing third order free variables occurring in x_1 or x_2"

$$\& \exists \mathbf{P}_1\, \forall \mathbf{P}_2\, \forall \mathbf{T} \left\{ \right.$$

$$\forall h\, \forall W \left\langle \left[\forall \ulcorner \mathbf{S} \urcorner\, \forall \ulcorner R \urcorner\, \forall y (\text{"}\mathbf{S} \text{ is a variable occurring in } x_1\text{"} \right.\right.$$

$$\& \text{ "}R \text{ is a variable occurring in } x_3\text{"} \;\&\; y = \ulcorner \mathbf{S}(R) \urcorner$$

$$\Rightarrow (\mathbf{T}(y, h, W) \equiv \exists T$$

$$\left.\left. (\forall z_1 \ \ldots \ \forall z_k\, (T(z_1, \ldots, z_k) \equiv W(\ulcorner R \urcorner, z_1, \ldots, z_k)) \,\&\, \mathbf{P}_1(\ulcorner \mathbf{S} \urcorner, T)))) \right]$$

$$\& \left[\text{an analogous clause for } x_2 \text{ and } \mathbf{P}_2\right]$$

[1] That is, $\ulcorner x \urcorner$ satisfies in $\mathbf{M}$ some fixed decent first order formula which FM-represents "being a first order variable".

$$\& \Big[\forall^{\ulcorner} v_1 {}^{\urcorner} \forall^{\ulcorner} v_2 {}^{\urcorner} \forall^{\ulcorner} v_3 {}^{\urcorner} \forall y \, (y = {}^{\ulcorner} v_1 + v_2 = v_3 {}^{\urcorner}$$
$$\Rightarrow (\mathbf{T}(y, h, W) \equiv h({}^{\ulcorner} v_1 {}^{\urcorner}) + h({}^{\ulcorner} v_2 {}^{\urcorner}) = h({}^{\ulcorner} v_3 {}^{\urcorner})))\Big]$$
$$\& \Big[\text{clauses for the remaining vocabulary symbols and logical connectives}]$$
$$\& \Big[\forall^{\ulcorner} v {}^{\urcorner} \forall y \, \forall y_1 \, (y = {}^{\ulcorner} \exists v {}^{\urcorner} \frown y_1)$$
$$\Rightarrow (\mathbf{T}(y, h, W) \equiv \exists h' \, (\forall z \neq {}^{\ulcorner} v {}^{\urcorner} \, (h'(z) = h(z)) \,\&\, \mathbf{T}(y_1, h', W))))\Big]$$
$$\& \Big[\text{the clause for the second order quantifier}]\Big\rangle$$
$$\Rightarrow \forall h \, \forall W \, \mathbf{T}(x_3, h, W)\Big\}\Big\rangle,$$

Clearly, the above formula is equivalent to a Σ^2_2 formula. It remains to explain why it defines FM-truth for $s[\Sigma^2_2]^{\leqslant k}$.

In all sufficiently large models, a given $s[\Sigma^2_2]^{\leqslant k}$ sentence x will be recognized as such and correctly decomposed into the third order prefix (consisting of the existential part (x_1) and universal part (x_2)) and the second order part (x_3, possibly containing third order variables which are quantified in the prefix). Now, x is true in a model if and only if there exists a valuation for the variables in x_1 (this valuation is represented by $\mathbf{P}_1$) such that for all valuations for the variables of x_2 (represented by $\mathbf{P}_2$), the second order part x_3 — with third order parameters interpreted according to $\mathbf{P}_1$ and $\mathbf{P}_2$ — is true.

The truth of x_3 is expressed using the third order variable $\mathbf{T}$. Call a third order relation $\mathbf{T}$ (of the appropriate type) *well-behaved* if $\mathbf{T}$ has the property that: for any first and second order valuations h and W and any *SO* formula y with all second order variables of arity $\leqslant k$ (but possibly with third order variables from among those in x_1 or x_2) recognized by the model as a formula of this type, $\mathbf{T}(y, h, W)$ holds if and only if y is true under the valuations $\mathbf{P}_1$, $\mathbf{P}_2$, h, and W. Our truth definition says that for any well-behaved $\mathbf{T}$ and any valuations h and W, $\mathbf{T}(x_3, h, W)$ holds, which in all sufficiently large models will be equivalent to the truth of x_3 (under the valuations given by $\mathbf{P}_1$ and $\mathbf{P}_2$).

Note that since we are using only decent FM-representing formulae, there is no danger of conflicting requirements being imposed on well-behaved relations — i.e., it cannot happen that some element of a model seems, according to the FM-representing formulae, to be, say, both some formula y and its negation. This is a technical but essential point, as otherwise $\mathbf{T}(y, h, W)$ would be required both to hold and not to hold for a well-behaved $\mathbf{T}$ and any

h, W (thus, no well-behaved relations could exist, and any $s[\Sigma_2^2]^{\leqslant k}$ sentence would satisfy the truth definition in the given model).

Bibliography

[Ajt83] M. Ajtai, Σ_1^1 *sentences on finite structures*, Annals of Pure and Applied Logic **24** (1983), 1–48.

[Ats02] A. Atserias, *Fixed-point logics, descriptive complexity, and random satisfiability*, Ph.D. thesis, University of California at Santa Cruz, 2002.

[BD01] J. van Benthem and K. Doets, *Higher order logic*, Handbook of Philosophical Logic. Second Edition (D. M. Gabbay and F. Guenther, eds.), vol. 1, Kluwer, 2001, pp. 189–244.

[BDG90] J. L. Balcázar, J. Díaz, and J. Gabarró, *Structural Complexity II*, Springer-Verlag, 1990.

[BDG95] ———, *Structural Complexity I. Second, Revised Edition*, Springer-Verlag, 1995.

[Ben62] J. H. Bennett, *On spectra*, Ph.D. thesis, Princeton, 1962.

[BFFT01] H. Buhrman, S. Fenner, L. Fortnow, and L. Torenvliet, *Two oracles that force a big crunch*, Computational Complexity **10** (2001), 93–116.

[BH91] S. Buss and L. Hay, *On truth-table reducibility to SAT*, Information and Computation **91** (1991), 86–102.

[Bör84] E. Börger, *Decision problems in predicate logic*, Logic Colloquium '82 (G. Lolli et al, ed.), North-Holland, 1984, pp. 263–301.

[BT94] H. Buhrman and L. Torenvliet, *On the cutting edge of relativization: The resource bounded injury method*, Proc. 21st International Colloquium on Automata, Languages and Programming,

Lecture Notes in Computer Science, vol. 820, Springer-Verlag, 1994, pp. 263–273.

[Chr74] C. A. Christen, *Spektren und Klassen Elementarer Funktionen*, Ph.D. thesis, ETH Zürich, 1974.

[CK03] S. A. Cook and A. Kolokolova, *A second-order system for polytime reasoning based on Grädel's theorem*, Annals of Pure and Applied Logic **124** (2003), 193–231.

[CKS81] A. K. Chandra, D. Kozen, and L. J. Stockmeyer, *Alternation*, Journal of the ACM **28** (1981), no. 1, 114–133.

[Coo71] S. Cook, *The complexity of theorem proving procedures*, Proc. 3rd ACM Symposium on Theory of Computing, ACM, 1971, pp. 151–158.

[DGH98] A. Dawar, G. Gottlob, and L. Hella, *Capturing relativized complexity classes without order*, Mathematical Logic Quarterly **44** (1998), 109–122.

[DH95] A. Dawar and L. Hella, *The expressive power of finitely many generalized quantifiers*, Information and Computation **123** (1995), 172–184.

[EF95] H. D. Ebbinghaus and J. Flum, *Finite Model Theory*, Springer-Verlag, 1995.

[EGG96] T. Eiter, G. Gottlob, and Y. Gurevich, *Normal forms for second-order logic over finite structures, and classification of NP optimization problems*, Annals of Pure and Applied Logic **78** (1996), 111–125.

[Fag74] R. Fagin, *Generalized first-order spectra and polynomial-time recognizable sets*, Complexity of Computation, SIAM-AMS Proceedings, vol. 7, 1974, pp. 43–73.

[Fag75] ———, *A spectrum hierarchy*, Zeitschrift für Mathematische Logik und Grundlagen der Mathematik **21** (1975), 123–134.

[FLZ92] B. Fu, H. Li, and Y. Zhong, *Some properties of exponential time complexity classes*, Proc. 7th Structure in Complexity Theory Conference, 1992, IEEE Computer Society Press, 1992, pp. 50–57.

[For94] L. Fortnow, *The role of relativization in complexity theory*, Bulletin of the EATCS **52** (1994), 229–243.

[GLV99] G. Gottlob, N. Leone, and H. Veith, *Succinctness as a source of complexity in logical formalisms*, Annals of Pure and Applied Logic **97** (1999), 231–260.

[Got97] G. Gottlob, *Relativized logspace and generalized quantifiers over finite ordered structures*, Journal of Symbolic Logic **62** (1997), 545–574.

[HCC+92] J. Hartmanis, R. Chang, S. Chiari, D. Ranjan, and P. Rohatgi, *Relativization: a revisionistic retrospective*, Bulletin of the EATCS **47** (1992), 144–153.

[Hem89] L. A. Hemachandra, *The strong exponential hierarchy collapses*, Journal of Computer and System Sciences **39** (1989), 299–322.

[Hem94] E. Hemaspaandra, *Census techniques collapse space classes*, Information Processing Letters **51** (1994), 85–92.

[Hen50] L. Henkin, *Completeness in the theory of types*, Journal of Symbolic Logic **15** (1950), 81–91.

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

[HS00] T. Hyttinen and G. Sandu, *Henkin quantifiers and the definability of truth*, Journal of Philosophical Logic **29** (2000), 507–527.

[HT03] L. Hella and J. M. Turull Torres, *Expressibility of higher order logics*, Electronic Notes in Theoretical Computer Science **84** (2003).

[Imm86] N. Immerman, *Relational queries computable in polynomial time*, Information and Control **68** (1986), 86–104.

[Imm87] ———, *Languages that capture complexity classes*, SIAM Journal on Computing **16** (1987), 760–778.

[Imm99] ———, *Descriptive Complexity*, Springer-Verlag, 1999.

[Kl04] A. Kolokolova, *Systems of bounded arithmetic from descriptive complexity*, Ph.D. thesis, University of Toronto, 2004.

[Koł04a] L. A. Kołodziejczyk, *A finite model-theoretical proof of a property of bounded query classes within PH*, Journal of Symbolic Logic **69** (2004), 1105–1116.

[Koł04b] ———, *Truth definitions in finite models*, Journal of Symbolic Logic **69** (2004), 183–200.

[KZ05] M. Krynicki and K. Zdanowski, *Theories of arithmetics in finite models*, Journal of Symbolic Logic **70(1)** (2005), 1–28.

[Lei87] D. Leivant, *Descriptive characterizations of computational complexity*, Journal of Computer and System Sciences **39** (1987), 51–83.

[Lei94] ———, *Higher order logic*, Handbook of Logic in Artificial Intelligence and Logic Programming (D. M. Gabbay et al, ed.), vol. 2, Oxford University Press, 1994, pp. 228–321.

[Lib04] L. Libkin, *Elements of Finite Model Theory*, Springer-Verlag, 2004.

[LL76] R. Ladner and N. Lynch, *Relativization of questions about log-space reducibility*, Mathematical Systems Theory **10** (1976), 19–32.

[Moc96] S. Mocas, *Separating classes in the exponential-time hierarchy from classes in PH*, Theoretical Computer Science **158** (1996), 221–231.

[Mos01] M. Mostowski, *On representing concepts in finite models*, Mathematical Logic Quarterly **47** (2001), 513–523.

[Mos03] ———, *On representing semantics in finite models*, Philosophical Dimensions of Logic and Science: Selected Contributed Papers from the 11th International Congress of Logic, Methodology, and Philosophy of Science, Krakow, 1999 (A. Rojszczak, J. Cachro, and G. Kurczewski, eds.), Kluwer, 2003, pp. 15–28.

[MP96] J. A. Makowsky and Y. B. Pnueli, *Arity and alternation in second order logic*, Annals of Pure and Applied Logic **78** (1996), 189–202.

[NIR03] A. Nash, R. Impagliazzo, and J. Remmel, *Universal languages and the power of diagonalization*, Proc. 18th IEEE Conference on Computational Complexity, IEEE Computer Society Press, 2003, pp. 337–346.

[Pap94] C. H. Papadimitriou, *Computational Complexity*, Addison Wesley, 1994.

[Qui70] W. V. O. Quine, *Philosophy of Logic*, Prentice-Hall, 1970.

[Raa00] P. Raatikainen, *The concept of truth in a finite universe*, Journal of Philosophical Logic **29** (2000), 617–633.

[Rus08] B. Russell, *Mathematical logic as based on the theory of types*, American Journal of Mathematics **30** (1908), 222–262.

[Sch93] J. R. Schoenfield, *Recursion Theory*, Springer-Verlag, 1993.

[ST95] A. P. Stolboushkin and M. A. Taitslin, *Is first order contained in an initial segment of $PTIME$?*, Computer Science Logic, 8th International Workshop, CSL '94, Lecture Notes in Computer Science, vol. 933, Springer-Verlag, 1995, pp. 242–248.

[Ste93] I. A. Stewart, *Logical characterization of bounded query classes I: logspace oracle machines*, Fundamenta Informaticae **18** (1993), 65–92.

[Sto77] L. Stockmeyer, *The polynomial-time hierarchy*, Theoretical Computer Science **3** (1977), 1–22.

[Tar33] A. Tarski, *Pojęcie prawdy w językach nauk dedukcyjnych*, Towarzystwo Naukowe Warszawskie, 1933, English version in [Tar56].

[Tar56] A. Tarski, *The concept of truth in formalized languages*, Logic, Semantics, Metamathematics (J. H. Woodger, ed.), Oxford University Press, 1956, translated from the German version by J. H. Woodger, pp. 152–278.

[Tra50] B. Trachtenbrot, *The impossibility of an algorithm for the decision problem for finite domains*, Doklady Akademii Nauk SSSR **70** (1950), 569–572, in Russian.

[Var82] M. Vardi, *The complexity of relational query languages*, Proc. 14th ACM Symposium on Theory of Computing, ACM, 1982, pp. 137–146.

[Wag90] K. W. Wagner, *Bounded query classes*, SIAM Journal on Computing **19** (1990), 833–846.

[Zda05] K. Zdanowski, *Arithmetics in finite but potentially infinite worlds*, Ph.D. thesis, Warsaw University, 2005, in preparation.

Index