Finite Model Theory

H.-D. Ebbinghaus
Freiburg

J. Flum
Freiburg

January 30, 1995
Preliminary version of some chapters

Draft

Contents

1 Descriptive Complexity Theory **2**
1.1 Some extensions of first-order logic 3
1.2 Turing Machines and Complexity Classes 7
1.2.1 Digression: Trahtenbrot's Theorem 9
1.2.2 Structures as Inputs 12
1.3 Logical descriptions of computations 16
1.4 The complexity of the satisfaction relation 30
1.5 The Main Theorem and some consequences 33

Draft

Chapter 1

Descriptive Complexity Theory

In the introduction to this book we gave the example of a database D that contains the names of the main cities in the world and the pairs (a, b) of cities such that a certain airline offers service from a to b without stopover. D may be interpreted as a first-order structure, more precisely, as a digraph $\mathcal{G} = (G, E^G)$, where G is the set of cities and $E^G ab$ means that there is a flight from a to b without stopover. Now, first-order logic may be viewed as a query language. For example, let

$$\varphi(x, y) \quad := \quad Exy \vee \exists z(Exz \wedge Ezy).$$

If φ is thought of as a query to D, the response will consist in the set of pairs (a, b) of cities such that a can be reached from b with at most one stop. First-order logic provides a rich class of database queries. However, there are plausible queries which are not first-order expressible. The query Q "Is it possible to fly from x to y using only the airline in question?" to databases of type D is an example. Also from a computational point of view answers to Q are more complex than answers to φ. Descriptive Complexity Theory analyzes the complexity class of all queries definable in a given logic, the central question being the following: Given a complexity class $\mathcal{C}$, is there a logic $\mathcal{L}$ such that the queries definable in $\mathcal{L}$ are precisely the queries in $\mathcal{C}$? For short: Given a complexity class $\mathcal{C}$, is there a logic that captures $\mathcal{C}$?

In the first section we briefly introduce some extensions of first-order logic which have been considered with this aim in mind. Their model theory will be studied in Chapter ??. The second section is devoted to complexity classes of structures. In particular, we describe how structures may be viewed as inputs to Turing machines. In section 1.3 we show how certain logics can be used to describe computations of a given complexity class. In section 1.4 we look

Draft

for the other direction and study the complexity of the satisfaction relation of such logics. By bringing together both sides we are lead to the descriptive characterizations of complexity classes we aim at, thus constituting a bridge between complexity theory and model theory. First consequences are given in the last section of this chapter.

1.1 Some extensions of first-order logic

We introduce inflationary fixed-point logic, partial fixed-point logic, deterministic transitive closure logic and transitive closure logic. These logics are obtained from first-order logic by adding operations well-suited to describe iterative and recursive procedures, for example, the behaviour of computing machines. As already remarked in the introduction to this chapter, their model theory will be studied in ??.

Let M be a finite nonempty set. Denote by $P(M)$ the power set of M, $P(M) := \{X \mid X \subseteq M\}$. A function $F : P(M) \to P(M)$ induces a sequence

$$\emptyset,\ F(\emptyset),\ F(F(\emptyset)), \ldots$$

of subsets of M. For its members we write $F_0, F_1, \ldots$, so $F_0 = \emptyset$ and $F_{n+1} = F(F_n)$. Suppose there is an $n_0 \in \mathbb{N}$ such that $F_{n_0+1} = F_{n_0}$, that is, $F(F_{n_0}) = F_{n_0}$. Then $F_m = F_{n_0}$ for all $m \geq n_0$. We denote F_{n_0} by F_∞ and say that the *fixed-point* F_∞ of F *exists*. In case the fixed-point F_∞ does not exist, we agree to set $F_\infty := \emptyset$.

F is said to be *inflationary*, if $X \subseteq M$ implies

$$X \subseteq F(X).$$

Lemma 1.1.1 (a) *If F_∞ exists, then $F_\infty = F_{2^{\|M\|}}$.*

(b) *If F is inflationary, then F_∞ exists and $F_\infty = F_{\|M\|}$.*

(c) *If F is arbitrary and $F' : P(M) \to P(M)$ is given by $F'(X) := X \cup F(X)$, then F' is inflationary.*

Proof. (a) Suppose F_∞ exists. As the power set $P(M)$ has $2^{\|M\|}$ elements, there are $m < 2^{\|M\|}$ and $l \geq 1$ such that $F_m = F_{m+l}$. If $l = 1$ then $F_m = F_{m+1}$, and hence, $F_m = F_{2^{\|M\|}} = F_\infty$. If $F_m \neq F_{m+1}$ then for $k \geq 0$ we have $F_{m+k \cdot l} \neq F_{m+k \cdot l+1}$, and hence F_∞ does not exist.

(b) By assumption, $F_0 \subseteq F_1 \subseteq \ldots \subseteq M$. Since M has $\|M\|$ elements, this sequence must get constant not later than with $F_{\|M\|}$. □

Let $\varphi(x_1, \ldots, x_k, \overline{u}, X, \overline{Y})$ be a formula in the vocabulary τ, where the relation variable X has arity k; moreover, let $\mathcal{A}$ be a τ-structure, $\overline{b}$ an interpretation of

Draft

$\overline{u}$ in $\mathcal{A}$, and $\overline{S}$ an interpretation of $\overline{Y}$ over A. Then $\varphi, \mathcal{A}, \overline{b}$, and $\overline{S}$ give rise to an operation $F^\varphi : P(A^k) \to P(A^k)$ $(F^\varphi = F^{\varphi, \mathcal{A}, \overline{b}, \overline{S}})$ defined by

$$F^\varphi(R) \quad := \quad \{(a_1, \ldots, a_k) \mid \mathcal{A} \models \varphi[a_1, \ldots, a_k, \overline{b}, R, \overline{S}]\}.$$

The notation F^φ does not make explicit relevant data; it reminds that F^φ can be viewed as a global relation. Note that for the function $(F^\varphi)'$ associated to F^φ as in part (c) of the preceding lemma we have $(F^\varphi)' = F^{(X\overline{x} \vee \varphi)}$.

To give an example, let $\mathcal{G} = (G, E^G)$ be a graph and

$$\varphi(x, y, X) \quad := \quad (x = y \vee \exists z(Xxz \wedge Ezy))$$

with xy corresponding to $\overline{x}$ above. Then for $n \geq 1$,

$$F^\varphi_n = \{(a, b) \mid \text{there is a path of length} \leq n \text{ from } a \text{ to } b\},$$

and hence,

$$F^\varphi_\infty = \{(a, b) \mid \text{there is a path from a to b}\}.$$

Inflationary Fixed-Point Logic FO(IFP) and *Partial Fixed-Point Logic* FO(PFP) are obtained by closing first-order logic FO under inflationary and arbitrary fixed-points of definable operations, respectively. We state the precise definitions.

For a vocabulary τ the class FO(IFP)$[\tau]$ of formulas of FO(IFP) of vocabulary τ is given by the calculus (we use the succinct notation

$$\frac{\varphi_1, \ldots, \varphi_n}{\varphi}$$

for the clause "If $\varphi_1, \ldots, \varphi_n$ are formulas then φ is a formula")

- $\overline{\varphi}$ $\quad \varphi$ an atomic second-order formula over τ
- $\frac{\varphi}{\neg\varphi}$; $\quad \frac{\varphi, \psi}{(\varphi \vee \psi)}$; $\quad \frac{\varphi}{\exists x \varphi}$
- $\frac{\varphi}{[\mathrm{IFP}_{\overline{x}, X}\varphi]\,\overline{t}}$ where the lengths of $\overline{x}$ and $\overline{t}$ are the same and coincide with the arity of X.

For FO(PFP) we replace the last rule by

- $\frac{\varphi}{[\mathrm{PFP}_{\overline{x}, X}\varphi]\,\overline{t}}$ where the lengths of $\overline{x}$ and $\overline{t}$ are the same and coincide with the arity of X.

Sentences are formulas without free first- and second-order variables, where the free occurrence of variables is defined in the standard way, adding for FO(IFP) the clause

$$\mathrm{free}\big([\mathrm{IFP}_{\overline{x}, X}\varphi]\,\overline{t}\big) \quad := \quad \mathrm{free}(\overline{t}) \cup (\mathrm{free}(\varphi) \setminus \{\overline{x}, X\}),$$

draft

and for FO(PFP) the clause

$$\mathrm{free}\big([\mathrm{PFP}_{\overline{x},X}\varphi]\,\overline{t}\big) \quad := \quad \mathrm{free}(\overline{t}) \cup (\mathrm{free}(\varphi) \setminus \{\overline{x}, X\}).$$

The semantics is defined inductively w.r.t. the above calculus, the meaning of $[\mathrm{IFP}_{\overline{x},X}\varphi]\,\overline{t}$ being "$\overline{t} \in F_\infty^{(X\overline{x}\vee\varphi)}$", and that of $[\mathrm{PFP}_{\overline{x},X}\varphi]\,\overline{t}$ being "$\overline{t} \in F_\infty^{\varphi}$". More precisely: If X is $k$ary and if the variables free in $[\mathrm{IFP}_{\overline{x},X}\varphi]\,\overline{t}$ are among $\overline{u}$ and $\overline{Y}$, and $\overline{b}$ and $\overline{S}$ are interpretations in $\mathcal{A}$ of $\overline{u}$ and $\overline{Y}$, respectively, then

$$\mathcal{A} \models [\mathrm{IFP}_{\overline{x},X}\varphi]\,\overline{t}\;[\overline{b},\overline{S}] \qquad \text{iff} \qquad (t_1[\overline{b}],\ldots,t_k[\overline{b}]) \in F_\infty^{(X\overline{x}\vee\varphi)},$$

$$\mathcal{A} \models [\mathrm{PFP}_{\overline{x},X}\varphi]\,\overline{t}\;[\overline{b},\overline{S}] \qquad \text{iff} \qquad (t_1[\overline{b}],\ldots,t_k[\overline{b}]) \in F_\infty^{\varphi}.$$

Examples 1.1.2 (a) In the language of graphs the formula of FO(IFP)

$$\psi_0(x,y) \quad := \quad [\mathrm{IFP}_{xy,X}\,(x = y \vee \exists z(Xxz \wedge Ezy))]\,xy$$

expresses that x, y are connected. Hence the class of connected graphs is axiomatizable in FO(IFP) by (the graph axioms and) $\forall x \forall y\, \psi_0(x,y)$.

(b) For $\tau = \{<, S, \min, \max\}$ the sentence of FO(IFP)

$$\neg[\mathrm{IFP}_{x,X}(x = \min \vee \exists y \exists z(Xy \wedge Syz \wedge Szx))]\max$$

together with the order axioms axiomatizes the class of orderings of even cardinality.

To compare the expressive power of logics we introduce the following relations.

Definition 1.1.3 Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be logics.

(a) $\mathcal{L}_1 \leq \mathcal{L}_2$ (read: $\mathcal{L}_1$ *is at most as expressive as* $\mathcal{L}_2$) if for every τ and every sentence $\varphi \in \mathcal{L}_1[\tau]$ there is a sentence $\psi \in \mathcal{L}_2[\tau]$ such that $\mathrm{Mod}(\varphi) = \mathrm{Mod}(\psi)$.

(b) $\mathcal{L}_1 \equiv \mathcal{L}_2$ (read: $\mathcal{L}_1$ and $\mathcal{L}_2$ have the same expressive power) if $\mathcal{L}_1 \leq \mathcal{L}_2$ and $\mathcal{L}_2 \leq \mathcal{L}_1$.

(c) $\mathcal{L}_1 < \mathcal{L}_2$ if $\mathcal{L}_1 \leq \mathcal{L}_2$ and not $\mathcal{L}_1 \equiv \mathcal{L}_2$.

In most cases, a proof of $\mathcal{L}_1 \leq \mathcal{L}_2$ even yields that every formula of $\mathcal{L}_1$ is equivalent to a formula of $\mathcal{L}_2$. If we want to emphasize this fact we say that $\mathcal{L}_1 \leq \mathcal{L}_2$ *holds even for formulas.*

Clearly, FO $\leq$ SO, where SO stands for second-order logic.

Proposition 1.1.4 FO(IFP) $\leq$ FO(PFP).

Draft

Proof. Note that $[\mathrm{IFP}_{\overline{x},X}\varphi]\,\overline{t}$ is equivalent to $[\mathrm{PFP}_{\overline{x},X}(X\overline{x} \vee \varphi)]\,\overline{t}$. □

Let R be a binary relation on a set M, $R \subseteq M^2$. A. The *transitive closure* $\mathrm{TC}(R)$ of R is defined as

$$\mathrm{TC}(R) := \{(a,b) \in M^2 \mid \text{there exist } n > 0 \text{ and } e_0, \ldots, e_n \in M \text{ such that } a = e_0, b = e_n, \text{ and for all } i < n,\ (e_i, e_{i+1}) \in R\}.$$

And the *deterministic transitive closure* $\mathrm{DTC}(R)$ is defined as

$$\mathrm{DTC}(R) := \{(a,b) \in M^2 \mid \text{there exist } n > 0 \text{ and } e_0, \ldots, e_n \in M \text{ such that } a = e_0, b = e_n, \text{ and for all } i < n,\ e_{i+1} \text{ is the unique } e \text{ for which } (e_i, e) \in R\}.$$

Transitive Closure Logic FO(TC) and *Deterministic Transitive Closure Logic* FO(DTC) are obtained by closing FO under the transitive closure and the deterministic transitive closure of definable relations, respectively. More precisely: For a vocabulary τ the class of formulas of FO(TC) of vocabulary τ is given by the calculus

- $\dfrac{}{\overline{\varphi}}$ where φ is an atomic first-order formula over τ
- $\dfrac{\varphi}{\neg\varphi}$; $\dfrac{\varphi, \psi}{(\varphi \vee \psi)}$; $\dfrac{\varphi}{\exists x \varphi}$
- $\dfrac{\varphi}{[\mathrm{TC}_{\overline{x},\overline{y}}\,\varphi]\,\overline{s}\overline{t}}$ where the variables in $\overline{x}, \overline{y}$ are pairwise distinct and where the tuples $\overline{x}, \overline{y}, \overline{s}$, and $\overline{t}$ are all of the same length, $\overline{s}$ and $\overline{t}$ being tuples of terms.

For FO(DTC) the last rule is replaced by $\dfrac{\varphi}{[\mathrm{DTC}_{\overline{x},\overline{y}}\,\varphi]\,\overline{s}\overline{t}}$ with the same side conditions. We define

$$\text{free}\,([\mathrm{TC}_{\overline{x},\overline{y}}\,\varphi]\,\overline{s}\overline{t}) = \text{free}\,([\mathrm{DTC}_{\overline{x},\overline{y}}\,\varphi]\,\overline{s}\overline{t}) := \text{free}\,(\overline{s}) \cup \text{free}\,(\overline{t}) \cup (\,\text{free}\,(\varphi) \setminus \{\overline{x}, \overline{y}\}).$$

The meaning of $[\mathrm{TC}_{\overline{x},\overline{y}}\,\varphi(\overline{x},\overline{y})]\,\overline{s}\overline{t}$ is "$(\overline{s},\overline{t}) \in \mathrm{TC}(\{(\overline{x},\overline{y}) \mid \varphi(\overline{x},\overline{y})\})$", and the meaning of $[\mathrm{DTC}_{\overline{x},\overline{y}}\,\varphi(\overline{x},\overline{y})]\,\overline{s}\overline{t}$ is "$(\overline{s},\overline{t}) \in \mathrm{DTC}(\{(\overline{x},\overline{y}) \mid \varphi(\overline{x},\overline{y})\})$" (cf. section ?? for details).

Thus a graph is connected, if it is a model of

$$\forall x \forall y(\neg x = y \to [\mathrm{TC}_{x,y} Exy]\,xy).$$

Proposition 1.1.5 (a) $\mathrm{FO(DTC)} \leq \mathrm{FO(TC)}$.

(b) $\mathrm{FO(TC)} \leq \mathrm{FO(IFP)}$.

Proof. For (a) note that the formulas $[\mathrm{DTC}_{\overline{x},\overline{y}}\varphi]\,\overline{s}\overline{t}$ and $[\mathrm{TC}_{\overline{x},\overline{y}}(\varphi(\overline{x},\overline{y}) \wedge \forall\overline{z}(\varphi(\overline{x},\overline{z}) \to \overline{z} = \overline{y}))]\,\overline{s}\overline{t}$ are equivalent, and for (b) that the same holds for the formulas $[\mathrm{TC}_{\overline{x},\overline{y}}\varphi(\overline{x},\overline{y})]\,\overline{s}\overline{t}$ and $[\mathrm{LFP}_{\overline{x}\,\overline{y},X}(\varphi(\overline{x},\overline{y}) \vee \exists\overline{u}(X\overline{x}\,\overline{u} \wedge \varphi(\overline{u},\overline{y})))]\,\overline{s}\overline{t}$. □

Draft

Definition 1.1.6 Let K be a class of τ-structures and $\mathcal{L}$ a logic. K is *axiomatizable* in $\mathcal{L}$, if there is a sentence of $\mathcal{L}$ of vocabulary τ such that $K = \mathrm{Mod}(\varphi)$.

When relating logics and complexity classes it is convenient to restrict to sufficiently large structures. This does not affect problems of axiomatizability. In fact, for a class K of structures and $m \geq 1$ denote by K_m the subclass of K of structures of cardinalitiy $\geq m$,

$$K_m \quad := \quad \{\mathcal{A} \mid \mathcal{A} \in K, \|A\| \geq m\}.$$

For every structure $\mathcal{A}$ there is a sentence $\varphi_{\mathcal{A}}$ of FO characterizing $\mathcal{A}$ up to isomorphism (cf. ??), that is,

$$\mathcal{B} \models \varphi_{\mathcal{A}} \text{ iff } \mathcal{B} \cong \mathcal{A}$$

holds for all $\mathcal{B}$. Hence, for any logic $\mathcal{L}$ with $FO \leq \mathcal{L}$,

$$K \text{ is axiomatizable in } \mathcal{L} \text{ iff } K_m \text{ is axiomatizable in } \mathcal{L}.$$

More exactly, setting $\varphi_m := \bigvee\{\varphi_{\mathcal{A}} \mid \mathcal{A} \in K, \|A\| < m, A = \{1, ..., \|A\|\}\}$ we have that $K = \mathrm{Mod}(\varphi)$ implies $K_m = \mathrm{Mod}(\varphi \wedge \neg\varphi_m)$, and $K_m = \mathrm{Mod}(\psi)$ implies $K = \mathrm{Mod}(\psi \vee \varphi_m)$.

1.2 Turing Machines and Complexity Classes

The aim of the present section is to recall basic definitions and results from computation theory, to fix our computation model for structures as inputs, and to introduce the corresponding complexity classes. It is not intended as a substitute for a course in computation theory, and some familiarity with the basic notions will be helpful. For more information, see, for example, [].

We adopt a computation model which belongs to the most popular ones in theoretical computer science, the Turing machine model. Our choice is motivated by the fact that Turing machine computations allow for simple descriptions and for natural definitions of complexity classes. However, as Turing machines deal only with strings, any other data must be coded by strings. The tedious work of encoding is one price we have to pay for the advantages above. Moreover, the model does not well reflect parallel, distributed, or real time computations, as only local changes are made in a step. However, these shortages are not really important for our aims. (Nevertheless, the future development of finite model theory migth eventually call for other computation models!)

For the following, we fix a finite alphabet $\mathbb{A}$. A Turing machine M is a finite device that performs operations on a tape which is bounded to the left and unbounded to the rigth and divided into *squares* (or *cells*). The machine operates stepwise, each step leading from one situation to a new one. In a situation every square of the tape either contains a single symbol from $\mathbb{A}$ or is

blank. In the letter case we say that it contains the symbol "blank". There is one exception: the leftmost or "virtual" cell always contains an endmark or "virtual" letter α which is not in $\mathbb{A}$. M has a reading/writing head which, in any situation, scans a single square of the tape and, in any step of a computation, erases or replaces the scanned symbol by another one and moves one cell to the left or to the right or remains at its place.

In each situation, M is in one of the states of a finite set State(M), the *set of states* of M. State(M) contains a special state s_0, the *initial* state, and special states s_+ and s_-, the *accepting* and the *rejecting* state, respectively. We assume that s_0, s_+ and s_- are pairwise distinct. The action or behaviour of M in a situation depends on the current state of M and on the symbol currently being scanned by the head. This dependance is given by Instr(M), the *set of instructions* of M. Each instruction has the form

$$(*) \qquad sa \to s'bh$$

where

- $s, s' \in \text{State}(M),\ s \neq s_+,\ s \neq s_-$
- $a, b \in \mathbb{A} \cup \{\alpha, \text{blank}\}$ and $(a = \alpha \text{ iff } b = \alpha)$
- $h \in \{-1, 0, 1\}$, and if $a = \alpha$ then $h \neq -1$.

The instruction $(*)$ means: If you are in state s and your head scans a cell with symbol a, replace a by b, move your head one cell to the left ($h = -1$), or right ($h = 1$) or don't move ($h = 0$); finally, change to state s'.

A machine M is *deterministic*, if for all $s \in S$ and $a \in \mathbb{A} \cup \{\text{blank}\}$ there is at most one instruction of the form $(*)$ in Instr(M).

As usual, denote by $\mathbb{A}^*$ the set of words over $\mathbb{A}$ and by $\mathbb{A}^+$ the set of nonempty words over $\mathbb{A}$. Let $u \in \mathbb{A}^*, u = a_1 \ldots a_r$ with $a_i \in \mathbb{A}$. M *starts* with u, if M begins a computation (or *run*) in state s_0 in the situation

BILD

The computation proceeds stepwise, each step corresponding to the execution of one instruction of M. The machine stops when it is in a state s scanning a symbol $a \in \mathbb{A} \cup \{\alpha, \text{blank}\}$ such that there is no instruction of the form $(*)$ in Instr(M). If $s = s_+$ we speak of an *accepting run*, if $s = s_-$, of a *rejecting run*. M *accepts* u, if there is at least one accepting run of M started with u, and M *rejects* u, if all runs are finite and rejecting.

Subsets of $\mathbb{A}^+$ are called *languages*. A language $L \subseteq \mathbb{A}^+$ is *accepted* by M, if for all $u \in \mathbb{A}^+$,

$$M \text{ accepts } u \qquad \text{iff} \qquad u \in L;$$

L is *decided* by M, if in addition

$$M \text{ rejects } u \qquad \text{iff} \qquad u \notin L.$$

Clearly, if M decides L then M accepts L. L is said to be *decidable*, if it is decided by some Turing machine and *acceptable* or *recursively enumerable*, if it is accepted by some M.

For a function $f : \mathbb{N} \to \mathbb{N}$ we say that M is *f time-bounded*, if for all $u \in \mathbb{A}^+$ accepted by M there is an accepting run of M started with u which has length at most $f(|u|)$ (here $|u|$ denotes the length of the word u). And M is *f space-bounded*, if for all $u \in \mathbb{A}^+$ accepted by M there is an accepting run which scans at most $f(|u|)$ cells before stopping.

Denote by $\mathbb{N}[x]$ the set of polynomials with natural coefficients. A language $L \subseteq \mathbb{A}^+$ is in PTIME ("polynomial time") or in PSPACE ("polynomial space"), if it is accepted by a deterministic machine that is p time-bounded or p space-bounded, respectively, for some polynomial $p \in \mathbb{N}[x]$. The classes NPTIME ("nondeterministic polynomial time") and NPSPACE ("nondeterministic polynomial space") are defined similarly, now allowing nondeterministic machines.

Immediately from the definitions one gets

$$\text{PTIME} \subseteq \text{NPTIME} \quad \text{and} \quad \text{PTIME} \subseteq \text{PSPACE} \subseteq \text{NPSPACE},$$

and one can show that

$$\text{NPTIME} \subseteq \text{PSPACE} \quad \text{and} \quad \text{PSPACE} = \text{NPSPACE}.$$

Hence

$$\text{PTIME} \subseteq \text{NPTIME} \subseteq \text{PSPACE} \ (= \text{NPSPACE}).$$

Let $\mathbb{A} = \{a, b\}$ and $L := \{u \in \mathbb{A}^+ \mid u \text{ contains an even number of } a's\}$. One can easily design a machine that accepts L and is time-bounded by the polynomial $x + 1$: the head just runs over the string, the state being "even" or "odd" depending on whether the number of $a's$ already scanned is even or odd, respectively. Essentially, this machine does not need any "working space" but only the space for the input. In order to measure only the working space and to introduce complexity classes like LOGSPACE and NLOGSPACE, where the working space needed is smaller than the input space, it is convenient to separate the input and to introduce machines with an input tape and a working tape. For other purposes it might be useful to introduce several input tapes and several working tapes, maybe also (if calculating a function, for example) one or more output tapes, with a head for each tape, where the heads can move independently from each other. It turns out that the definition of the usual complexity classes does not depend on the number of tapes or on other peculiarities such as the form of the tape, i.e. whether it is unbounded to both sides or not. We shall often use this independence, choosing, for example, the number of input and working tapes according to needs and to convenience.

1.2.1 Digression: Trahtenbrot's Theorem

We have already remarked that first-order logic, when restricted to the finite, does not admit a complete proof calculus. In the following we give a proof of

Trahtenbrot's Theorem, which immediately implies this fact (cf.).

Fix an alphabet $\mathbb{A}$. It is well-known that is not decidable whether a deterministic Turing machine M accepts the empty word, by short, whether M *halts* ("Undecidability of the Halting Problem").[1] We shall define a vocabulary $\sigma(\mathbb{A})$ and then, in an effective way, assign to every deterministic machine M over $\mathbb{A}$ a sentence φ_M of $FO[\sigma(\mathbb{A})]$ such that

$(*)$ $\qquad\qquad$ φ_M is satisfiable in the finite iff M halts.

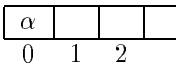
This shows

Theorem 1.2.1 (Trahtenbrot's Theorem) *Finite Satisfiability is not decidable, that is, the set*

$$Sat[\sigma(\mathbb{A})] := \{\varphi \mid \varphi \text{ a sentence of } \mathrm{FO}[\sigma(\mathbb{A})] \text{ satisfiable in the finite}\}$$

is not decidable.

Proof. Without loss of generality we can restrict ourselves to deterministic Turing machines whose set of states is an initial segment $\{0, \ldots, s_M\}$ of the natural numbers, $s_0 := 0$ being the initial state and 1 the accepting state. We number the cells of the tape as indicated by



In particular, the number 0 is given to the virtual cell. If the machine M makes at least n steps, we let C_n denote the n-th *configuration*. C_n contains the following data: the state, the number of the cell scanned by the head, and the tape inscription after n steps. If M stops after n steps in a state distinct from the accepting state, we set $C_{n+1} = C_n$, $C_{n+2} = C_n, \ldots$. Let $\tau_0 = \{<, S, \min, \max\}$ be the vocabulary for orderings. In the following we sometimes write 0 instead of min. In addition to the symbols of τ_0 the vocabulary $\sigma(\mathbb{A})$ contains relation symbols State (binary), Head (binary), and for any $a \in \mathbb{A} \cup \{\alpha, \text{blank}\}$ a binary relation symbol Letter_a.

For every natural number $n \geq s_M$ (recall that $\{0, \ldots, s_M\}$ is the set of states) we define a structure $\mathcal{A}_n$ with universe $\{0, \ldots, n\}$, which reflects the initial segment $C_0, \ldots, C_n$ of the computation of M started with the empty word (however, only $C_0, \ldots, C_k$ with $k < n$, if M stops in the accepting state after k steps). For $s, t \leq n$

$\text{State}^{A_n} s\, t$	iff	according to C_t, the state is s
$\text{Head}^{A_n} i\, t$	iff	according to C_t, the head is in cell i
$\text{Letter}_a^{A_n} i\, t$	iff	according to C_t, the letter in cell i is a.

The sentence φ_M will have the properties (a) and (b), which immediately give the equivalence $(*)$ stated above:

[1] More exactly: One encodes Turing machines M in some natural way as words w_M. Then the language $\{w_M \mid M \text{ halts}\}$ is not decidable.

(a) If M, started with the empty word, stops after k steps in the accepting state) and $n \geq s_M, k$, then $\mathcal{A}_n \models \varphi_M$.

(b) If $\mathcal{A}$ is a finite model of φ_M and M, started with the empty word, runs at least k steps, then $\|A\| \geq k$.

As φ_M we take the conjunction of the $\{<, S, \min, \max\}$-order axioms together with the conjunction of the following sentences (1)-(5) (where we write 0 for min, 1 for the successor of min, etc.):

(1) "The univere has at least $s_M + 1$ elements."

(2) $\text{State}\, 0\, 0 \wedge \text{Head}\, 1\, 0 \wedge \text{Letter}_\alpha 0\, 0 \wedge \forall x(\neg x = 0 \to \text{Letter}_{\text{blank}}\, x\, 0)$

(at time 0, the state is 0, the head is in cell 1, the virtual cell contains α and all other cells are empty).

(3) For each instruction $sa \to s'bh$ a conjunct $\varphi_{sa\to s'bh}$, which describes the changes due to this instruction. If $h = 0$ then $\varphi_{sa\to s'bh}$ is the sentence

$$\begin{aligned}&\forall y \forall t \Big((\text{State}\, s\, t \wedge \text{Head}\, y\, t \wedge \text{Letter}_a\, y\, t) \\ &\quad \to \exists t' (S t\, t' \wedge \text{State}\, s'\, t' \wedge \text{Head}\, y\, t' \wedge \text{Letter}_b\, y\, t' \\ &\wedge \forall v (\neg v = y \to \bigwedge_{a \in \mathbb{A} \cup \{\alpha, \text{blank}\}} (\text{Letter}_a\, v\, t \to \text{Letter}_a\, v\, t'))) \Big).\end{aligned}$$

We leave it to the reader to write down the sentence $\varphi_{sa\to s'bh}$ for $h = -1$ and $h = 1$.

(4) For any sa such that there is no instruction of the form $sa \to s'bh$ for M and where s is a state distinct from the accepting state s_+ $(= 1)$ a conjunct φ_{sa} that corresponds to our convention $C_{m+1} = C_m$, $C_{m+2} = C_m, \ldots$ (if according to C_m, M is in state s scanning a):

$$\begin{aligned}\varphi_{sa} := \forall y \forall t \Big(&(\text{State}\, s\, t \wedge \text{Head}\, y\, t \wedge \text{Letter}_a\, y\, t) \to \exists t' (S\, t\, t' \wedge \text{State}\, s\, t' \\ &\wedge \text{Head}\, y\, t' \wedge \forall v \bigwedge_{a \in \mathbb{A} \cup \{\alpha, \text{blank}\}} (\text{Letter}_a\, v\, t \to \text{Letter}_a\, v\, t')) \Big).\end{aligned}$$

(5) $\exists t\ \text{State}\, 1\, t$

(the accepting state is reached).

Coding $\sigma(\mathbb{A})$-structures as graphs (cf. **??**) one obtains, from the undecidability of $\text{Sat}[\sigma(\mathbb{A})]$, that for a binary relation symbol E the set

$$\text{Graph-Sat} := \{\varphi \mid \varphi \text{ an FO}[\{E\}] \text{ sentence satisfiable in a finite graph}\}$$

is not decidable. Hence, for any vocabulary τ containing an at least binary relation symbol, the set

$$\text{Sat}[\tau] := \{\varphi \mid \varphi \text{ an FO}[\tau] \text{ sentence satisfiable in the finite}\}$$

Draft

is not decidable.

Clearly, there is a decision procedure that, for any finite structure $\mathcal{A}$ whose universe is a set of natural numbers and for any first-order sentence φ, checks whether $\mathcal{A} \models \varphi$. Since φ is satisfiable in the finite iff there is a model $\mathcal{A}$ of φ with $A = \{0, \ldots, n\}$ for some n, we can use this decision procedure to enumerate $\mathrm{Sat}[\tau]$. Together with Trahtenbrot's Theorem this shows:

Theorem 1.2.2 *If τ contains an at least binary relation symbol then the set*

$$\mathrm{Val}[\tau] := \{\varphi \in \mathrm{FO}[\tau] \mid \varphi \textit{ a sentence valid in all finite structures}\}$$

of sentences valid in all finite structures is not enumerable.

Proof. For any sentence φ we have

$$(+) \qquad \varphi \notin \mathrm{Sat}[\tau] \quad \text{iff} \quad \neg\varphi \in \mathrm{Val}[\tau].$$

If $\mathrm{Val}[\tau]$ would be enumerable, then, by (+), $\mathrm{FO}[\tau] \setminus \mathrm{Sat}[\tau]$ would be enumerable, too. This would lead to a decision procedure for $\mathrm{Sat}[\tau]$ (contradicting Trahtenbrot's Theorem): Given a sentence φ, start enumeration procedures for $\mathrm{Sat}[\tau]$ and $\mathrm{FO}[\tau] \setminus \mathrm{Sat}[\tau]$ until one of them yields φ. □

Now we have the negative result concerning the proof calculus: If there would be a complete proof calculus for first-order logic in the finite, we could effectively enumerate all possible formal proofs and hence also the sentences valid in the finite, contradicting Theorem 1.2.2.

1.2.2 Structures as Inputs

We fix the conventions which allow to regard finite structures as inputs to Turing machines. Note that, in general, structures are abstract objects and the same holds for their elements, so that there is no canonical way of representing structures by strings (or by sequences of strings). If the same structure has different representations, the machine should produce the same answer on all the representations. First we make life easier by restricting attention to ordered structures. To do so we first recall the definition (cf. ??).

Definition 1.2.3 Let $\{<\} \subseteq \tau_0 \subseteq \{<, S, \min, \max\}$ and $\tau_0 \subseteq \tau$. A τ-structure $\mathcal{A}$ is *ordered*, if the reduct $\mathcal{A}|\tau_0$ is an ordering (that is, $<^A$ is an ordering and S, min and max, if present, are interpreted by the successor relation, the least and the last element of the ordering, respectively). $\mathcal{O}(\tau)$ is the class of ordered τ-structures.

Let $\mathcal{A} \in \mathcal{O}(\tau)$ be an ordered structure with $\|A\| = n$. By passing to an isomorphic copy we can assume – and always will tacitly do so – that $A = \{0, \ldots, n-1\}$ and that $<^A$ is the natural ordering on this set; that is, we identify or "label"

the least element of $<^A$ in $\mathcal{A}$ with 0, its successor with 1 etc. $\mathcal{A}$ is sometimes called a *labeled* structure.

Suppose $\tau = \tau_0 \cup \tau_1$. For representational purposes we think of τ_1 as a tuple $(R_1, \ldots, R_k, c_1, \ldots, c_l)$ rather than a set.

A *Turing machine for τ-structures* will have $1+k+l$ *input tapes* and m *work tapes* for some $m \geq 1$. All tapes are bounded to the left and unbounded to the right. Their cells are numbered as indictated by[2]

BILD

BILD

The "virtual cell" numbered with -1 always contains the virtual letter α. All input tapes will contain an input word followed by the virtual letter ω indicating the end of the input word. Each tape has its own head. The heads can move independently from each other. Those on input tapes are *read-only* heads (i.e. they do not change the scanned letter), while those on the work tapes are *read-and-write* heads. The alphabet only contains the symbol "1". Moreover, we identify "0" with "blank", that is, during a computation almost all cells contain 0.

With an ordered τ-structure $\mathcal{A}$ we associate the following input inscriptions on the $1 + k + l$ input tapes (numbered from 0 to $k + l$): The 0-th tape, the "universe tape", contains a sequence of $1's$ of length $n := \|A\|$,

BILD

For $1 \leq i \leq k$, the i-th input tape contains the information about $R := R_i$ coded as follows: Suppose R is r-ary, that is, $R^A \subseteq \{0, \ldots, n-1\}^r$. Of course, $\|\{0, \ldots, n-1\}\|^r = n^r$. For $j < n^r$ let $|j|_r$ be the j-th r-tuple in the lexicographic order of $\{0, \ldots, n-1\}^r$; in other words, look at the unique n-adic representation of j,

$$j = j_1 n^{r-1} + j_2 n^{r-2} + \ldots + j_{r-1} n + j_r \quad \text{with} \quad 0 \leq j_i < n,$$

and set $|j|_r := (j_1, \ldots, j_r)$. (If n is not clear from the context, we write $|j|_r^n$ instead of $|j|_r$). Then the i-th input tape has the inscription

BILD

where

$$a_j = 1 \qquad \text{iff} \qquad R^A |j|_r$$

(and, hence, $a_j = 0$ iff not $R^A |j|_r$).

For $1 \leq i \leq l$, the $(k + i)$-th input tape contains the binary representation of $j := c_i^A$ without leading zeros.

We say that a Turing machine M is *started with* $\mathcal{A}$, if the input tapes contain

[2]For the purposes of the next sections this numbering is more appropriate than that of the section on Trahtenbrot's Theorem, where we numbered the virtual cell by 0.

Draft

the information on $\mathcal{A}$ in the way just described, the work tapes are empty, and each head scans the cell numbered 0 of its tape. As in the case of one-tape machines, M has finite sets State(M) of states and Instr(M) of instructions. State(M) contains an *initial* (*or starting*) state s_0, an *accepting* state s_+ and a *rejecting* state s_-. *Instructions* now have the form

$$(*) \qquad s\, b_0 \ldots b_{k+l}\, c_1 \ldots c_m \to s' c'_1 \ldots c'_m\, h_0 \ldots h_{k+l+m}$$

with the meaning

> if you are in state s, your heads scan $b_0, \ldots, b_{k+l}$ on the input tapes and $c_1, \ldots, c_m$ on the work tapes, replace $c_1, \ldots, c_m$ by $c'_1 \ldots c'_m$, respectively, move the i-th head according to h_i and, finally, change to state s'.

Here, $s,\ s' \in$ State(M), $b_0, \ldots, b_{k+l},\ c_1, \ldots, c_m,\ c'_1 \ldots c'_m \in \{0, 1, \alpha, \omega\}$, and $h_0, \ldots, h_{k+l+m} \in \{-1, 0, 1\}$. Moreover,

- if $b_j = \alpha$ then $h_j \neq -1$
  (if the head scans the leftmost square it cannot move left)
- if $b_j = \omega$ then $h_j \neq 1$
- if $c_j = \alpha$ then $h_{k+l+j} \neq -1$ and $c'_j = \alpha$
- if $c_j \in \{0, 1\}$ then $c'_j \in \{0, 1\}$
- $s \neq s_+$ and $s \neq s_-$.

The *base* of the instruction in $(*)$ is given by

$$s\, b_0, \ldots, b_{k+l}\, c_1, \ldots, c_m$$

M is said to be *deterministic*, if no two distinct instructions in Instr(M) have the same base. Sometimes, to emphasize that we do not require a machine to be deterministic, we speak of a *nondeterministic* machine.

For a function $f : \mathbb{N} \to \mathbb{N}$ we say that M is f *time-bounded*, if for any $\mathcal{A}$ accepted by M there is an accepting run of M, started with $\mathcal{A}$, of length at most $f(\|A\|)$. And M is f *space-bounded*, if for all $\mathcal{A}$ accepted by M there is an accepting run which scans at most $f(\|A\|)$ squares on each work tape before stopping.

Let K be a class of ordered τ-structures. M *accepts* K, if M accepts exactly those ordered τ-structures that lie in K. For classes of structures the definitions of PTIME ("polynomial time"), NPTIME ("nondeterministic polynomial time"), PSPACE ("polynomial space") are introduced in the obvious way. For example,

Draft

- K is in PTIME iff there is a deterministic machine M and a polynomial $p \in \mathbb{N}[x]$ such that M accepts K and M is p time-bounded.

And we define

- K is in NLOGSPACE, "nondeterministic logarithmic space" (LOGSPACE, "deterministic logarithmic space") iff there is a (deterministic) machine M and $d \geq 1$ such that M accepts K and is $d \cdot \log$ space-bounded ($\log\, n$ stands for the least natural number $\geq \log_2 n$).

Exercise 1.2.4 Denote by $l(\mathcal{A})$ the sum of the lengths of the input words on the input tapes of a machine started with $\mathcal{A}$. Show that in the definitions of the complexity classes mentioned above we can replace $\|A\|$ by $l(\mathcal{A})$.

At the end of section **??**, for a class K of ordered structures and $m \geq 1$, we have observed that the class

$$K_m := \{\mathcal{A} \in K \mid \|A\| \geq m\}$$

is axiomatizable in a logic $\mathcal{L}$ iff K is axiomatizable in $\mathcal{L}$. Analogously, for any of the complexity classes $\mathcal{C}$ introduced so far, we have

$$K \in \mathcal{C} \qquad \text{iff} \qquad K_m \in \mathcal{C}$$

(since we can change a machine in such a way that it runs on a given finite set of inputs in a prescribed form without essentially changing its time and space requirements; cf. Exercise 1.2.5 below). We give an application of this fact. Suppose that K is in PSPACE. Then K_2 is in PSPACE, too, say, accepted by a machine M that is q space-bounded, where $q(x) = a_s x^s + a_{s-1} x^{s-1} + \ldots + a_1 x + a_0$. For suitable d, $q(n) \leq n^d$ for all $n \geq 2$. Thus M is x^d space-bounded. These observations will allow us to restrict to monic polynomials $p(x) = x^d$ when considering PSPACE (or PTIME, NPTIME).

Exercise 1.2.5 For fixed τ and $m \geq 1$ there is $c_m \in \mathbb{N}$ such that for any $K \subseteq \mathcal{O}(\tau)$ and any $f : \mathbb{N} \to \mathbb{N}$:

(i) If K is accepted by an f time-bounded machine, then K_m is accepted by an $f + (m+1)$ time-bounded machine, where $f + (m+1)$ denotes the function whose value for n is $f(n) + (m+1)$.

(ii) If K_m is accepted by an f time-bounded machine, then K_m is accepted by an $f + c_m$ time-bounded machine.

(iii) K is accepted by an f space-bounded machine iff K_m is accepted by an f space-bounded machine.

1.3 Logical descriptions of computations

Let K be a class of ordered τ-structures, $K \subseteq \mathcal{O}(\tau)$. We write $K \in$ IFP, if K is axiomatizable in FO(IFP), and use similar notations for the other logics. Our main goal is to show

- $K \in$ LOGSPACE iff $K \in$ DTC
- $K \in$ NLOGSPACE iff $K \in$ TC
- $K \in$ PTIME iff $K \in$ IFP
- $K \in$ NPTIME iff $K \in \Sigma^1_1$
- $K \in$ PSPACE iff $K \in$ PFP

(Σ^1_1 denotes the fragment of second-order logic consisting of the sentences of shape $\exists X_1 \ldots \exists X_m \psi$, where ψ is first-order, cf. section ??).

These results provide the bridge between logic and complexity we have aimed at by relating purely machine-oriented characterizations and characterizations by means of logical definability. Section 1.5 exemplifies the methodological possibilities they offer.

In this section we prove the implications from left to right, the other implications will be settled in the next section.

Let $\mathcal{C}$ be one of the complexity classes above and $\mathcal{L}$ the logic associated to $\mathcal{C}$ by the corresponding equivalence. Assume that $K \in \mathcal{C}$, and let M be a Turing machine witnessing that $K \in \mathcal{C}$. We are going to describe the behaviour of M by a formula φ_M of $\mathcal{L}$ in such a way that for any ordered structure $\mathcal{A}$,

$$\mathcal{A} \models \varphi_M \quad \text{iff} \quad M \text{ accepts } \mathcal{A},$$

and hence,

$$K = \text{Mod}(\varphi_M).$$

The reader familiar with classical recursion theory will observe the analogy of the proofs given below with usual proofs showing the μ-recursiveness of Turing computable functions. There, the transition from one configuration to the next is coded by "simple" (e.g. primitive recursive) functions and the μ-operator is used to detect the end of a computation. Similarly here, using the ordering relation we can describe the transition from one configuration to the next by means of first-order logic. The additional expressive power, for example the operation IFP, is used to find out whether the computation stops and to get its outcome. In recursion theory the proof of the equivalence shows that one application of the μ-operator suffices. Similarly we shall obtain, say, that every formula of FO(IFP) is equivalent to a formula with only one occurrence of IFP.

For the following we fix a vocabulary $\tau = \tau_0 \cup \tau_1$, where, for simplicity, we

Draft

assume that $\tau_0 = \{<, S, \min, \max\}$ and τ_1 is relational, $\tau_1 = (R_1, \ldots, R_k)$ with r_i-ary R_i .

A Turing machine M for τ-structures has $1+k$ input tapes and a certain number m of work tapes. In order to describe computations that M performs when started with a structure, we introduce *configurations* which contain the relevant data of possible situations of a computation. These data are:

- the current state
- the current inscriptions of the work tapes
- the current position of the heads on both the input and the work tapes.

An *accepting configuration* is a configuration with control state s_+. A configuration CONF′ is said to be a *successor configuration* of CONF, if an instruction of M allows M to pass from CONF to CONF′ in one step. For convenience, we extend this definition by saying that any *accepting* configuration is a successor configuration of itself.[3] If M is deterministic then every configuration has at most one successor configuration.

Let M be a (nondeterministic!) Turing machine for structures, which is x^d space-bounded, that is, if M accepts an ordered structure $\mathcal{A}$, then there is an accepting run that scans at most n^d squares on every work tape, where $n := \|A\|$. We can assume that $r_i \leq d$ for $i = 1, \ldots, k$ (r_i being the arity of R_i). Fix a structure $\mathcal{A}$. By the remarks at the end of the last two sections when proving that a class of structures is axiomatizable in a logic or acceptable by a Turing machine of a certain complexity bound, we can restrict ourselves to sufficiently large finite structures. Here we look at structures $\mathcal{A}$ such that $\|A\| = n > k+m$ and $> \|\mathrm{State}(M)\|$. We assume that $\mathrm{State}(M)$, the set of states of M, is an initial segment of the natural numbers and $s_0 = 0$ is the starting state.

Let CONF be a configuration, where at most the n^d first cells of each work tape are not empty and where the heads scan one of these cells. A first attempt to code the contents of these cells could consist in dividing the relevant part of each work tape into $\frac{n^d}{\log n} =: r$ blocks of length $\log n$ and reading each block as a natural number $< n$ in binary representation. This would require variables $x_1, \ldots, x_r$ for each tape. Then a formula bearing the information on successive configurations would contain at least the variables $x_1, \ldots, x_r$ and would, therefore, depend on the cardinality n of the universe. We overcome this difficulty for PTIME, NPTIME and PSPACE by using relation variables instead of variables.

To code the data of CONF we first introduce the "state relation" $\mathrm{ST}^{\mathrm{CONF}}$, the "end-of-tape relations" $\mathrm{E}_j^{\mathrm{CONF}}$, the "head relations" $\mathrm{H}_j^{\mathrm{CONF}}$ and the "inscription relations" $\mathrm{I}_j^{\mathrm{CONF}}$. The last ones are only introduced for the work tapes,

[3] This convention differs from the corresponding one in the section on Trahtenbrot's Theorem.

since the inscriptions of input tapes are given by the input structure and kept fixed during the whole computation.

$$\mathrm{ST}^{\mathrm{CONF}} := \{s\}, \text{ where } s \text{ is the state of CONF}$$

for $0 \leq j \leq k+m$,

$$\mathrm{E}_j^{\mathrm{CONF}} := \begin{cases} \{0\}, & \text{if the } j\text{-th head faces } \alpha \\ \{n-1\}, & \text{if the } j\text{-th head faces } \omega \\ \emptyset, & \text{otherwise} \end{cases}$$

for $0 \leq j \leq k$,[4]

$$\mathrm{H}_j^{\mathrm{CONF}} := \{|e|_{r_j} \mid \text{the } j\text{-th head scans the } e\text{-th square and this does not contain } \alpha \text{ or } \omega\}.$$

For $k+1 \leq j \leq k+m$,

$$\mathrm{H}_j^{\mathrm{CONF}} := \{|e|_d \mid \text{the } j\text{-th head scans the } e\text{-th square and this does not contain } \alpha\}$$

for $k+1 \leq j \leq k+m$,

$$\mathrm{I}_j^{\mathrm{CONF}} := \{|e|_d \mid 0 \leq e < n^d \text{ and the } e\text{-th square of the } j\text{-th work tape contains the symbol 1}\}.$$

Obviously, CONF is uniquely determined by these relations. The starting configuration CONF_0, for example, is given by

$$\mathrm{ST}^{\mathrm{CONF}_0} = \{0\}, \quad \mathrm{E}_j^{\mathrm{CONF}_0} = \emptyset, \quad \mathrm{H}_j^{\mathrm{CONF}_0} = \{(0,\ldots,0)\}, \text{ and } \mathrm{I}_j^{\mathrm{CONF}_0} = \emptyset.$$

For technical convenience we encode CONF in a single $(d+2)$-ary relation $C^{\mathrm{CONF}} \subseteq \{0,\ldots,n-1\}^{d+2}$ by joining the preceding relations as follows (add two first coordinates to distinguish these relations and fill up with zeroes in the middle to get arity $(d+2)$):[5]

$$\begin{aligned} C^{\mathrm{CONF}} \quad := \quad & \{(0,0)\} \times \{\tilde{0}\} \times \mathrm{ST}^{\mathrm{CONF}}) \\ \cup \quad & \bigcup_{0 \leq j \leq k+m} (\{(1,j)\} \times \{\tilde{0}\} \times \mathrm{E}_j^{\mathrm{CONF}}) \\ \cup \quad & \bigcup_{0 \leq j \leq k+m} (\{(2,j)\} \times \{\tilde{0}\} \times \mathrm{H}_j^{\mathrm{CONF}}) \\ \cup \quad & \bigcup_{k+1 \leq j \leq k+m} (\{(3,j)\} \times \mathrm{I}_j^{\mathrm{CONF}}). \end{aligned}$$

[4] Recall that $|e|_{r_j}$ is the r_j-tuple given by the n-adic representation of e.

[5] Recall our convention that $\tilde{c}$ denotes the constant sequence $c \ldots c$.

draft

Clearly, given $C \subseteq \{0,\dots,n-1\}^{d+2}$, we can easily decide whether there is a configuration CONF of M, where only the first n^d cells of each work tape are relevant, such that $C = C^{\mathrm{CONF}}$, or as we shortly say, whether C is an n^d-bounded configuration.

The following lemma now states how the behaviour of M can be described by suitable formulas.

Lemma 1.3.1 *Let M be a Turing machine for τ-structures, which is x^d space-bounded. There is a formula $\varphi_{\mathrm{start}}(\overline{x}) \in \mathrm{FO}[\tau]$ and there are formulas $\varphi_{\mathrm{succ}}(\overline{x}, X)$ and $\psi_{\mathrm{succ}}(X, Y)$ in "$\mathrm{FO}[\tau]$" (that is, second-order formulas without second-order quantifiers) such that for all sufficiently large $\mathcal{A}$ and $\overline{a} \in A^{d+2}$ we have:*

(a) *"$\varphi_{\mathrm{start}}(\overline{x})$ describes the starting configuration": If C_0 denotes the starting configuration of M started with $\mathcal{A}$, then*

$$\mathcal{A} \models \varphi_{\mathrm{start}}[\overline{a}] \qquad \textit{iff} \qquad \overline{a} \in C_0.$$

(b) *"$\varphi_{\mathrm{succ}}(\overline{x}, X)$ describes the successor configuration of X": If M is deterministic and C is an n^d-bounded configuration of M (where $n := \|A\|$), then*

$$\mathcal{A} \models \varphi_{\mathrm{succ}}[\overline{a}, C] \;\; \textit{iff} \;\; C \textit{ has an } n^d\textit{-bounded successor configuration } C' \textit{ and } \overline{a} \in C'.$$

(c) *"$\psi_{\mathrm{succ}}(X, Y)$ expresses that Y is a successor configuration of X": If C_1 is an n^d-bounded configuration of M and C_2 a further $(d+2)$-ary relation on A, then*

$$\mathcal{A} \models \psi_{\mathrm{succ}}[C_1, C_2] \;\; \textit{iff} \;\; C_2 \textit{ is an } n^d\textit{-bounded configuration of } M, \textit{ which is a successor configuration of } C_1.$$

We postpone the proof of this lemma and first draw some consequences.

Theorem 1.3.2 *Let $K \subseteq \mathcal{O}(\tau)$ be a class of ordered structures. If K is in* PSPACE, *then K is axiomatizable in* FO(PFP).

Proof. Let M be a deterministic machine witnessing $K \in$ PSPACE. By previous remarks we can assume that M is x^d space-bounded for suitable d. We set

$$\varphi(\overline{x}, X) \quad := \quad (\neg\exists\overline{x} X\overline{x} \wedge \varphi_{\mathrm{start}}(\overline{x})) \vee (\exists\overline{x} X\overline{x} \wedge \varphi_{\mathrm{succ}}(\overline{x}, X)),$$

where φ_{start} and φ_{succ} are the formulas associated to M by the preceding lemma. Let $\mathcal{A}$ be an ordered structure and $n := \|A\|$. Note that $F_0^{\varphi}, F_1^{\varphi}, F_2^{\varphi}, \dots$ is the sequence $\emptyset, C_0, C_1, \dots$, where

- C_0 is the starting configuration
- if C_i is an n^d-bounded configuration of M with an n^d-bounded successor configuration C, then $C_{i+1} = C$. In particular, if C_i is accepting then $C_i = C_{i+1} = C_{i+2} \ldots$
- if C_i is an n^d-bounded configuration without successor configuration or with a successor configuration, which is not n^d-bounded, then $C_{i+1} = \emptyset$, $C_{i+2} = C_0$, $C_{i+3} = C_1, \ldots$, that is, the sequence has no fixed-point.

Summarizing, we have,

$$F^{\varphi}_{\infty} \text{ exists} \qquad \text{iff} \qquad M \text{ accepts } \mathcal{A},$$

and, if M accepts $\mathcal{A}$, then F^{φ}_{∞} is the corresponding accepting configuration. Therefore,

$$\begin{array}{lll} \mathcal{A} \in K & \text{iff} & M \text{ accepts } \mathcal{A} \\ & \text{iff} & \mathcal{A} \models \exists \overline{x} [\mathrm{PFP}_{\overline{x},X} \varphi] \overline{x}, \end{array}$$

that is, $K = \mathrm{ordMod}(\exists \overline{x}\, [\mathrm{PFP}_{\overline{x},X} \varphi]\, \overline{x})$ [6]. □

Corollary 1.3.3 *Let $K \subseteq \mathcal{O}(\tau)$ be in* PSPACE*. Then K is axiomatizable by a sentence of* FO(PFP) *with only one occurrence of* PFP*.* □

We now turn to PTIME and will realize that we can do better. Whereas accepting runs of polynomially space-bounded machines may have exponential length, we here only have to consider runs of polynomial length. This enables us to code the whole run into a single relation which can be obtained as the fixed-point of an inflationary process. On the logical side we therefore can replace PFP by IFP.

Consider a (finite or infinite) run $C_0, C_1, \ldots$ of an x^d time- (and, hence, x^d space-) bounded machine started with a structure of cardinality n. If the run accepts the structure, C_{n^d-1} must be an accepting configuration. The inflationary process indicated above is given by a formula φ such that

$$F^{\varphi}_i = \bigcup_{\substack{m < i \\ C_m \text{defined}}} \{|m|_d\} \times C_m, \quad \text{and} \quad F^{\varphi}_{\infty} = \bigcup_{\substack{m < n^d \\ C_m \text{defined}}} \{|m|_d\} \times C_m,$$

that is, we use the first d coordinates as time stamps when coding the run in one relation (as above, $|m|_d$ denotes the m-th tuple of $\{0, \ldots, n-1\}^d$).

Theorem 1.3.4 *Let $K \subseteq \mathcal{O}(\tau)$ be a class of ordered structures. If K is in* PTIME*, then K is axiomatizable in* FO(IFP)*.*

[6] Recall that $\mathrm{ordMod}(\psi)$ is the class of models of ψ whose τ_0-reduct is an ordering, or equivalently, $\mathrm{ordMod}(\psi)$=$\mathrm{Mod}(\psi \wedge \psi_0)$, where ψ_0 is the conjunction of the order axioms for the vocabulary τ_0.

Draft

Proof. Let M be a deterministic machine witnessing $K \in$ PTIME. We can assume that, for suitable d, M is x^d time-bounded. For $\overline{v} = v_0 \dots v_{d-1}$ we set

$$\varphi(\overline{v}, \overline{x}, Z) \quad := \quad (\overline{v} = \widetilde{\min} \wedge \varphi_{\text{start}}(\overline{x})) \vee \exists \overline{u}(S^d \overline{uv} \wedge \varphi_{\text{succ}}(\overline{x}, Z\overline{u}_-)).$$

Here, $\overline{v} = \widetilde{\min}$ abbreviates $v_0 = \min \wedge \dots \wedge v_{d-1} = \min$; $S^d \overline{uv}$ stands for "$\overline{v}$ is the successor of $\overline{u}$ in the lexicographic order", and $\varphi_{\text{succ}}(\overline{x}, Z\overline{u}_-)$ is obtained from $\varphi_{\text{succ}}(\overline{x}, X)$ by replacing subformulas $X\overline{t}$ by $Z\overline{u}\overline{t}$. Then we have:

T accepts $\mathcal{A}$
iff the state of the $(n^d - 1)$-th configuration of M, started with $\mathcal{A}$, is s_+
iff $\mathcal{A} \models \exists y(\text{“}y \text{ is the } s_+\text{-th element of } <\text{”} \wedge [\text{IFP}_{\overline{v}\,\overline{x},Z}\varphi]\widetilde{\max} \min \min \widetilde{\min} y)$

or succinctly,

$$\text{iff} \quad \mathcal{A} \models [\text{IFP}_{\overline{v}\,\overline{x},Z}\varphi]\widetilde{\max} \min \min \widetilde{\min} s_+$$

that is, K is the class of ordered models of a sentence of FO(IFP). □

Corollary 1.3.5 *Let $K \subseteq \mathcal{O}(\tau)$ be in* PTIME. *Then K is axiomatizable in* FO(IFP) *by a sentence with only one occurrence of* IFP. □

It is now easy to extend the previous result to NPTIME.

Theorem 1.3.6 *Let $K \subseteq \mathcal{O}(\tau)$ be a class of ordered structures. If K is in* NPTIME, *then K is axiomatizable in* SO *by a Σ^1_1 sentence (that is, by a sentence of the form $\exists X_1 \dots \exists X_r \psi$, where ψ contains no second-order quantifiers).*

Proof. Choose M witnessing $K \in$ NPTIME and assume that M is x^d time-bounded. Then, for $\mathcal{A} \in \mathcal{O}(\tau)$ with $n := \|A\|$,

$\mathcal{A} \in K$ iff there is a run of M, started with $\mathcal{A}$, of length $\leq n^d$ that accepts $\mathcal{A}$
iff there is a sequence $C_0, \dots, C_{n^d-1}$ of (n^d-bounded) configurations of M, started with $\mathcal{A}$, such that C_0 is the starting configuration, C_{i+1} is a successor configuration of C_i, and s_+ is the state of C_{n^d-1}
iff $\mathcal{A} \models \varphi$,

where φ is the sentence (the intended meaning of the second-order variable Z being $\bigcup_{m < n^d} \{|m|_d\} \times C_m$):

$$\varphi \quad := \quad \exists Z(\forall \overline{x}(Z\widetilde{\min}\overline{x} \leftrightarrow \varphi_{\text{start}}(\overline{x})) \wedge \forall \overline{u}\forall \overline{v}(S^d \overline{u}\,\overline{v} \rightarrow \psi_{\text{succ}}(Z\overline{u}_-, Z\overline{v}_-) \wedge Z\widetilde{\max} \min \min \widetilde{\min} s_+).$$

□

We now come back to a proof of the key Lemma 1.3.1.

Draft

Proof of 1.3.1. Let M be an x^d space-bounded machine for τ-structures. Recall the specific coding of an n^d-bounded configuration CONF in a single relation C^{CONF} comprising relations $\mathrm{ST}^{\mathrm{CONF}}, \mathrm{E}_j^{\mathrm{CONF}}, \mathrm{H}_j^{\mathrm{CONF}}$, and $\mathrm{I}_j^{\mathrm{CONF}}$ that contain the information on the state, the endmarks, the head positions, and the inscription of the work tapes. Let $\overline{x}$ be the sequence of variables $xyx_1 \dots x_d$.

For (a) we can set

$$\begin{array}{lcll} \varphi_{\mathrm{start}}(\overline{x}) & := & \overline{x} = \tilde{0} & \text{“the state is } s_0\text{”} \\ & & \vee\ (x = 2 \wedge 0 \leq y \leq k+m \wedge x_1 \dots x_d = \tilde{0}) & \text{“the heads scan the 0-th cell”} \end{array}$$

where, for example, $0 \leq y \leq k+m$ means that y is equal or less than the $(k+m)$-th element in the ordering $<$ and $x_d = 0$ stands for $x_d = \min$. Also in the following we shall use such selfexplanatory abbreviations.

For parts (b) and (c) we first introduce, for every instruction $\mathrm{instr} \in \mathrm{Instr}(M)$,

$$\mathrm{instr} = sb_0 \dots b_{k+l} c_1 \dots c_m \to s' c'_1 \dots c'_m h_0 \dots h_{k+l+m},$$

a formula $\varphi_{\mathrm{instr}}(\overline{x}, X)$ which for n^d-bounded configurations X expresses

> “X has base $s\overline{b}\overline{c}$ and if the successor configuration according to instr is not n^d-bounded, then $\{\overline{x} \mid \varphi_{\mathrm{instr}}(\overline{x}, X)\} = \emptyset$, otherwise $\{\overline{x} \mid \varphi_{\mathrm{instr}}(\overline{x}, X)\}$ is this successor configuration”.

Moreover, we need a formula $\varphi_{\mathrm{acc}}(X)$ which for n^d-bounded configurations X expresses

> “X is an accepting configuration”.

Using $\varphi_{\mathrm{instr}}(\overline{x}, X)$ and $\varphi_{\mathrm{acc}}(X)$, we get the desired formulas φ_{succ} and ψ_{succ} of parts (b) and (c) as (recall that we agreed to set $C_{m+1} = C_m$ for accepting configurations C_m)

$$\varphi_{\mathrm{succ}}(\overline{x}, X) \quad := \quad (\varphi_{\mathrm{acc}}(X) \wedge X\overline{x}) \vee \bigvee_{\mathrm{instr} \in \mathrm{instr}(M)} \varphi_{\mathrm{instr}}(\overline{x}, X),$$

and

$$\begin{array}{lcl} \psi_{\mathrm{succ}}(X, Y) & := & (\varphi_{\mathrm{acc}}(X) \wedge \forall \overline{x}(Y\overline{x} \leftrightarrow X\overline{x})) \vee \\ & & \bigvee_{\mathrm{instr} \in \mathrm{instr}(M)} (\exists \overline{x}\, \varphi_{\mathrm{instr}}(\overline{x}, X) \wedge \forall \overline{x}(Y\overline{x} \leftrightarrow \varphi_{\mathrm{instr}}(\overline{x}, X))). \end{array}$$

It remains to give $\varphi_{\mathrm{instr}}(\overline{x}, X)$ and $\varphi_{\mathrm{acc}}(X)$. We set

$$\varphi_{\mathrm{acc}}(X) \quad := \quad X00\tilde{0}s_+.$$

The formula $\varphi_{\mathrm{instr}}(\overline{x}, X)$ has the form

$$\varphi_{\mathrm{instr}}(\overline{x}, X) \quad := \quad (\varphi_{s, \overline{b}, \overline{c}}(X) \wedge \varphi_{s', \overline{c}', \overline{h}}(\overline{x}, X)),$$

Draft

where, for a n^d-bounded configuration X, the formula $\varphi_{s,\overline{b},\overline{c}}(X)$ expresses

"X has base $s, \overline{b}, \overline{c}$",

and $\varphi_{s',\overline{c}',\overline{h}}(\overline{x}, X)$ expresses

> "if the successor configuration Y of X according to $s', \overline{c}', \overline{h}$ is not n^d-bounded, then $\{\overline{x} \mid \varphi_{s',\overline{c}',\overline{h}}(\overline{x}, X)\} = \emptyset$, else $\{\overline{x} \mid \varphi_{s',\overline{c}',\overline{h}}(\overline{x}, X)\} = Y$".

For easier reading of the formulas below, we introduce the following abbreviations:

ENDMARK $yz := X1y\tilde{0}z$	"the y-th head faces the endmark z (0 for α and $n-1$ for ω)"
HEAD $y\overline{z} := X2y\tilde{0}\overline{z}$	"the y-th head is on position $\lvert\overline{z}\rvert$"
ONE $y\overline{z} := X3y\overline{z}$	"the y-th work tape contains 1 on position $\lvert\overline{z}\rvert$."

Now we take as $\varphi_{s,\overline{b},\overline{c}}(X)$ the conjunction of

$X00\tilde{0}s$

"s is the state"

$\bigwedge_{b_j=\alpha}$ ENDMARK j min $\wedge$ $\bigwedge_{c_j=\alpha}$ ENDMARK $(k+j)$ min

"heads at the left end of a tape"

$\bigwedge_{b_j=\omega}$ ENDMARK j max

"heads at the right end on input tapes "

$\bigwedge_{b_j=1} \exists x_1 \ldots \exists x_{r_j}(\text{HEAD } j\tilde{0}x_1 \ldots x_{r_j} \wedge R_j x_1 \ldots x_{r_j})$

"heads of input tapes facing a 1"

$\bigwedge_{b_j=0} \exists x_1 \ldots \exists x_{r_j}(\text{HEAD } j\tilde{0}x_1 \ldots x_{r_j} \wedge \neg R_j x_1 \ldots x_{r_j})$

"heads of input tapes facing a 0"

$\bigwedge_{c_j=1} \exists x_1 \ldots x_d\,(\text{HEAD } (k+j)x_1 \ldots x_d \wedge \text{ONE } (k+j)x_1 \ldots x_d)$

"heads of work tapes facing a 1"

$\bigwedge_{c_j=0} \exists x_1 \ldots x_d\,(\text{HEAD } (k+j)x_1 \ldots x_d \wedge \neg\text{ONE } (k+j)x_1 \ldots x_d)$

"heads of work tapes facing a 0".

Finally, we take as $\varphi_{s',\overline{c}',\overline{h}}(\overline{x}, X)$ the conjunction $(\varphi_1 \wedge \varphi_2)$, where φ_1 is

$$\bigwedge_{\substack{h_j=1 \\ k+1\le j\le k+m}} \neg\text{HEAD } j\widetilde{\max}$$

Draft

"heads of work tapes moving to the right do not face the (n^d-1)-th square"

and where φ_2 is the disjunction of

$(x = y = 0 \wedge x_1 \dots x_{d-1} = \tilde{0} \wedge x_d = s')$

"s' is the new state"

$$\bigvee_{k+1 \le j \le k+m} (\neg \text{HEAD } jx_1 \dots x_d \wedge \text{ONE } jx_1 \dots x_d \wedge x = 3 \wedge y = j)$$

"work tape content unchanged on squares not scanned"

$$\bigvee_{c'_j = 1} (\text{HEAD } jx_1 \dots x_d \wedge x = 3 \wedge y = k + j)$$

"new content 1 on scanned squares of a work tape"

$$\bigvee_{h_j = 1} (\text{ENDMARK } j0 \wedge x = 2 \wedge y = j \wedge x_1 \dots x_d = \tilde{0})$$

"heads scanning α and moving to the right come to position 0"

$$\bigvee_{h_j = 0} (\text{ENDMARK } j0 \wedge x = 1 \wedge y = j \wedge x_1 \dots x_d = \tilde{0})$$

"unchanged position of heads facing α"

$$\bigvee_{h_j = -1} (\text{ENDMARK } j \max \wedge x = 2 \wedge y = j \wedge x_1 \dots x_{d-r_j} = \tilde{0} \wedge x_{d-r_j+1} \dots x_d = \widetilde{\max})$$

"heads scanning ω and moving to the left come to position $n^d - 1$"

$$\bigvee_{h_j = 0} (\text{ENDMARK } j \max \wedge x = 1 \wedge y = j \wedge x_1 \dots x_{d-1} = \tilde{0} \wedge x_d = \max)$$

"unchanged position of heads facing ω"

$$\bigvee_{h_j = -1} (\text{HEAD } j\tilde{0} \wedge x = 1 \wedge y = j \wedge x_1 \dots x_d = \tilde{0})$$

"heads scanning α from their new position"

$$\bigvee_{\substack{j \le k \\ h_j = 1}} (\text{HEAD } j\tilde{0} \underbrace{\max \dots \max}_{r_j - \text{times}} \wedge x = 1 \wedge y = j \wedge x_1 \dots x_{d-1} = \tilde{0} \wedge x_d = \max)$$

"heads of input tapes scanning ω from their new position"

$$\bigvee_{j \le k+m} \exists u_1 \dots \exists u_d (\text{“}x_1 \dots x_d = y_1 \dots y_d + h_j\text{”} \wedge \text{HEAD } ju_1 \dots u_d \wedge x = 2 \wedge y = j)$$

"new head position of heads on 'interior' squares."

□

Finally, we come to the corresponding results for LOGSPACE and NLOGSPACE. Recall that $K \in$ NLOGSPACE means that there is a (nondeterministic) machine M and some $d \ge 1$ such that M accepts K and is $d \cdot \log$ space-bounded.

Every natural number $i < n$ codes a word over $\{0,1\}$ of length $\log n$, namely its binary representation $|i|^2_{\log n}$ of length $\log n$. Thus, using d variables, a number independent of the n, we can represent the relevant contents of a work tape. Moreover, by restricting to sufficiently large structures $\mathcal{A}$ we can assume that $d \cdot \log n \leq n$ (where $n := \|A\|$). Hence, the head positions can be represented by a single number $< n$. Altogether we can describe the data of a configuration by a sequence of natural numbers $< n$ of length indepent of n, where we agree to use the first number to represent the state. The exact definitions will be given later, when proving the following lemma.

Lemma 1.3.7 *Let M be $d \cdot \log$ space-bounded. Then there are formulas $\chi_{\text{start}}(\overline{x})$ of* FO *and $\chi_{\text{succ}}(\overline{x}, \overline{x}')$ of* FO(DTC) *such that for all sufficiently large $\mathcal{A}$ and $\overline{a}$ in A,*

(a) $\mathcal{A} \models \chi_{\text{start}}[\overline{a}]$ *iff $\overline{a}$ is the (description of the) starting configuration.*

(b) *For any $d \cdot \log \|A\|$-bounded configuration $\overline{a}$ and any $\overline{b}$,*

$$A \models \chi_{\text{succ}}[\overline{a}, \overline{b}] \quad \textit{iff} \quad \overline{b} \textit{ is a } d \cdot \log \|A\| \textit{-bounded successor configuration of } \overline{a}.$$

Before we give a proof we derive some consequences.

Theorem 1.3.8 *Let $K \subseteq \mathcal{O}(\tau)$ be a class of ordered structures. If $K \in$* LOGSPACE *then K is axiomatizable in* FO(DTC)*.*

Proof. Let M be a deterministic machine witnessing $K \in$ LOGSPACE. Let χ_{start} and χ_{succ} be the formulas associated to M according to the preceding lemma. Then, by (a) and (b) of this lemma, we have:

M accepts $\mathcal{A}$
iff there is a sequence $\overline{a}_0, \ldots, \overline{a}_k$ of $d \cdot \log \|A\|$-bounded configurations such that $\overline{a}_0$ is the starting configuration, $\overline{a}_{i+1}$ is the successor configuration of $\overline{a}_{i+1}$, and $\overline{a}_k$ is an accepting configuration
iff $\mathcal{A} \models \exists \overline{x}(\chi_{\text{start}}(\overline{x}) \wedge \exists \overline{x}'([\text{DTC}_{\overline{x},\overline{x}'} \chi_{\text{succ}}(\overline{x}, \overline{x}')]\overline{x}\,\overline{x}' \wedge x'_1 = s_+)$.

Hence, K is the class of ordered models of a sentence of FO(DTC). □

Theorem 1.3.9 *Let $K \subseteq \mathcal{O}(\tau)$ be a class of ordered structures. If $K \in$* NLOGSPACE *then K is axiomatizable in* FO(TC)*.*

Proof. Let M be a machine witnessing $K \in$ NLOGSPACE. Since, in general, M is nondeterministic we just have to replace DTC by TC in the last proof. In fact, we have

M accepts $\mathcal{A}$
iff $\mathcal{A} \models \exists \overline{x}(\chi_{\text{start}}(\overline{x}) \wedge \exists \overline{x}'([\text{TC}_{\overline{x},\overline{x}'} \chi_{\text{succ}}(\overline{x}, \overline{x}')]\overline{x}\overline{x}'. \wedge x'_1 = s_+)$.

Since FO(DTC) $\leq$ FO(TC), we have obtained a sentence of FO(TC) axiomatizing K. □

Denote by FO(posTC) the class of formulas of FO(TC), which only contain positive occurrences of TC, that is, each such occurrence is in the scope of an even number of negation symbols. In **??** we will see that FO(DTC) $\leq$ FO(posTC). Thus the preceding proof yields

Corollary 1.3.10 *If a class of ordered structures is in* NLOGSPACE*, then it is axiomatizable by a sentence of* FO(posTC)*.* □

We turn to a proof of Lemma 1.3.7. As already remarked above in the case of logspace bounded machines we code configurations not by relations but by numbers. This change requires that some arithmetical functions and predicates are definable in FO(DTC). The next lemma serves this purpose.

Lemma 1.3.11 *There are formulas* $\varphi_+(x,y,z)$, $\varphi_\cdot(x,y,z)$, $\varphi_2(x,y)$ *and* $\varphi_{\log(\text{universe})}(x)$ *such that for any ordered structure* $\mathcal{A}$ *with* $A = \{0,\ldots,\|A\|-1\}$ *and any* $a,b,c \in A$*,*

$$\begin{array}{lll} \mathcal{A} \models \varphi_+[a,b,c] & \textit{iff} & a+b=c \\ \mathcal{A} \models \varphi_\cdot[a,b,c] & \textit{iff} & a\cdot b=c \\ \mathcal{A} \models \varphi_2[a,b] & \textit{iff} & 2^a=b \\ \mathcal{A} \models \varphi_{\log(\text{universe})}[a] & \textit{iff} & a=\log\|A\|. \end{array}$$

Proof. For better readibility, instead of describing the natural numbers in terms of the ordering, we use constants $1, 2, \ldots$.

Given numbers x and y, we have the following path from $(0,x)$ to $y, x+y)$

$$(0,x) \to (1,x+1) \to (2,x+2) \to \ldots \to (y,x+y),$$

which shows that as $\varphi_+(x,y,z)$ we can take the formula

$$[\mathrm{DTC}_{uv,u'v'}\,(Suu' \wedge Svv')]\min xyz.$$

Similarly, the path

$$(1,x) \to (2,2x) \to (3,3x) \to \ldots \to (y, y\cdot x)$$

shows that we can set

$$\varphi_\cdot(x,y,z) \quad := \quad [\mathrm{DTC}_{u,v,u',v'}\,(\varphi_+(u,x,u') \wedge Svv')]1xyz$$

and the path

$$(0,1) \to (1,2) \to (2,4) \to \ldots \to (x,2^x)$$

that we can set

$$\varphi_2(x,y) \quad := \quad [\mathrm{DTC}_{u,v,u'v'}\,(Suu' \wedge \varphi_\cdot(v,2,v'))]\min 1xy.$$

Draft

Finally, let

$$\varphi_{\log(\text{universe})}(x) \quad := \quad \neg\exists y\varphi_2(x,y) \wedge \forall z(z < x \to \exists y\varphi_2(z,y)).$$

□

Write l for $\log n$. Recall that for m with $m < 2^{l+1}$ and $m_0, \ldots, m_l \in \{0,1\}$ we have

$$|m|^2_{l+1} = m_0 \ldots m_l \quad \text{iff} \quad m = m_0 \cdot 2^l + m_1 \cdot 2^{l-1} + \ldots + m_{l-1} \cdot 2 + m_l.$$

We then say that m_k is the k-th digit of $|n|^2_{l+1}$. If n, and hence l, is clear from the context, we denote $|m|^2_{l+1}$ by $[m]$.

Lemma 1.3.12 *There are formulas of* FO(DTC) *which, in ordered structures* $\mathcal{A}$*, define the following relations (where* $n := \|A\|$*,* $l = \log n$ *and* $m, k, u_1, \ldots, u_d, u'_1, \ldots, u'_d < n$*) :*

One mk	*iff*	$k < l+1$ *and the* k*-th digit of* $[m]$ *is 1*
Zero mk	*iff*	$k < l+1$ *and the* k*-th digit of* $[m]$ *is* 0
$\text{One}_d\ u_1 \ldots u_d k$	*iff*	$k < d \cdot (l+1)$ *and the* k*-th digit of the concatenation* $[u_1] \ldots [u_d]$ *is* 1
$\text{Zero}_d\ u_1 \ldots u_d k$	*iff*	$k < d \cdot (l+1)$ *and the* k*-th digit of the concatenation* $[u_1] \ldots [u_d]$ *is* 0
$\text{Equal}_d\ u_1 \ldots u_d k u'_1 \ldots u'_d$	*iff*	$k < d \cdot (l+1)$ *and the words* $[u_1] \ldots [u_d]$ *and* $[u'_1] \ldots [u'_d]$ *differ at most at the* k*-th position.*

We denote the corresponding formulas by $\varphi_{\text{one}}(x,z)$, $\varphi_{\text{zero}}(x,z)$, $\varphi_{d-\text{one}}(x_1, \ldots, x_d, z)$, $\varphi_{d-\text{zero}}(x_1, \ldots, x_d, z)$, and $\varphi_{d-\text{equal}}(\overline{x}, z, \overline{x}')$, respectively.

Proof. Note that for $m < 2^{l+1}$ and $k = 0, 1, \ldots$ the k-th digit of $[m]$ is 1 iff

$$\exists y \in \mathbb{N} \exists z \in \mathbb{N}(m = y \cdot 2^{l-k+1} + z \ \wedge\ 2^{l-k} \leq z < 2^{l-k+1}).$$

Using this fact one easily shows, for example, the following equivalences, which immediately can be formalized with the help of the of the formulas of the preceding lemma.

$\text{One}\, mk$	iff	$(k = 0 \wedge 2^l \leq m) \vee (1 \leq k \leq l \wedge \exists y \exists z(m = y \cdot 2^{l-k+1} + z \ \wedge\ 2^{l-k} \leq z < 2^{l-k+1}))$
$\text{One}_2\, u_1 u_2 k$	iff	$(k < l+1 \wedge \text{One}\, u_1 k) \vee (l+1 \leq k < 2(l+1) \ \wedge \text{One}\, u_2(k-(l+1)))$
$\text{Equal}_d\, \overline{u} k \overline{u}'$	iff	$k < d \cdot (l+1) \wedge \forall i((i < d \cdot (l+1) \wedge i \neq k) \to (\text{One}_d\, \overline{u} i \leftrightarrow \text{One}_d\, \overline{u}' i)))$.

□

Finally, we come to the

Proof of 1.3.7. Let M be a logarithmically space-bounded machine for τ-structures, say, M is $d \cdot \log n$ space-bounded. For simplicity, we assume that $\tau = \{R\}$ with a binary R and assume that M has only one work tape. We restrict ourselves to structures of cardinality n greater than $d \cdot \log n$ and greater than $s_M + 1$, where $\{0, \ldots, s_M\}$ is the set of states of M. When M is started with a structure $\mathcal{A}$, where $A = \{0, \ldots, n-1\}$, we can code the data of a resulting configuration by a tuple

$$(z, u_\alpha, u_\omega, u, v_\alpha, v_\omega, v_0, v_1, w_\alpha, w, y_1, \ldots, y_d)$$

where

- z is the state
- u_α, u_ω, u code the position of the head on the 0-th input tape (the "universe tape") by

$$u_\alpha = \begin{cases} 0 & \text{if the head does not face } \alpha \\ n-1 & \text{if the head faces } \alpha \end{cases}$$

$$u_\omega = \begin{cases} 0 & \text{if the head does not face } \omega \\ n-1 & \text{if the head faces } \omega \end{cases}$$

  and u is the number of the cell faced by the head, if it is an interior one, otherwise $u = 0$
- similarly, $v_\alpha, v_\omega, v_0, v_1$ code the position of the head on the first input tape, the tape for the binary relation R; the variables v_0, v_1 represent the head position $v_0 n + v_1$
- w_α, w code the position of the head on the work tape (note that $d \cdot \log n < n$)
- the concatenation $[y_1] \ldots [y_d]$ is the inscription of the first $d \cdot \log n$ cells of the work tape.

Sometimes, for notational simplicity, we denote the sequence $z u_\alpha u_\omega \ldots y_d$ simply by $\overline{x}$.

Concerning part (a) of the lemma, we can set

$$\chi_{\text{start}}(\overline{x}) \quad := \quad \overline{x} = \tilde{0}.$$

Concerning part (b), we define $\chi_{\text{succ}}(\overline{x}, \overline{x}')$ by

$$\chi_{\text{succ}}(\overline{x}, \overline{x}') \quad := \quad \chi_{\text{acc}}(\overline{x}, \overline{x}') \vee \bigvee_{\text{instr in } M} \chi_{\text{instr}}(\overline{x}, \overline{x}')$$

where

$$\chi_{\text{acc}}(\overline{x}, \overline{x}') \quad := \quad (x_1 = s_+ \wedge \overline{x}' = \overline{x})$$

Draft

(which in case $\overline{x}$ is a configuration expresses that $\overline{x}$ is accepting and $\overline{x}' = \overline{x}$) and where for every instruction

$$\text{instr} = sb_0b_1c_1 \to s'c_1'h_0h_1h_2$$

$\chi_{\text{instr}}(\overline{x}, \overline{x}')$ is a formula which, in case $\overline{x}$ is a $d \cdot \log n$ space-bounded configuration, expresses that $\overline{x}$ has base $sb_0b_1c_1$, the successor configuration of $\overline{x}$ according to instr is $d \cdot \log n$ space-bounded and is $\overline{x}'$. As an example, we explicitly give $\chi_{\text{instr}}(\overline{x}, \overline{x}')$ for

$$\text{instr} \;=\; s1\alpha 1 \to s'0(-1)11,$$

namely as the conjunction of

$z = s$

"s is the state"

$u_\alpha = \min \wedge u_\omega = \min$

"the head of the 0-th input tape faces an interior cell"

$v_\alpha = \max \wedge v_\omega = \min \wedge v = \min$

"the head of the first input tape faces α "

$w_\alpha = \min \wedge \ \text{One}_d\, y_1 \ldots y_d w$

"the head of the work tape faces a 1"

$z' = s'$

"s' is the new state"

$u_\omega' = \min \wedge ((u > 0 \wedge Su'u \wedge u_\alpha = \min) \ \vee (u = 0 \wedge u' = 0 \wedge u_\alpha = \max))$

"new head position of the 0-th input tape"

$v_\alpha = \min \wedge v = \min \wedge v_\omega = \min$

"new head position of the first input tape is cell 0"

$w_\alpha = \min \wedge Sww' \wedge w' < d \cdot \log n,$
i.e $w_\alpha = \min \wedge Sww' \wedge \exists x(\varphi_{\log(\text{universe})}(x) \wedge w' < x)$

"new head position of the work tape is within the bounds"

$\text{Zero}_d\, y_1' \ldots y_d' w$

"new content of cell scanned on the work tape"

$\text{Equal}_d\, y_1 \ldots y_d w y_1' \ldots y_d'$

"work tape content unchanged on cells not scanned".

□

Draft

1.4 The complexity of the satisfaction relation

Suppose, for example, that the class K of ordered structures is axiomatizable by the FO(IFP) sentence φ, $K = \{\mathcal{A} \in \mathcal{O}(\tau) \mid \mathcal{A} \models \varphi\}$. We aim at showing that $K \in$ PTIME, that is, for fixed φ we want to prove that the satisfaction relation $\mathcal{A} \models \varphi$ can be decided in time polynomially bounded in $\|A\|$. To simplify the corresponding algorithms we start with some general remarks. Note that manipulations in algorithms of the kind we now describe do not destroy polynominial time and logarithmic space bounds.

(1) Using an additional work tape W' it is possible at any time of a computation to move the head of a given work tape W to the rigthmost square which the head of the tape has scanned so far. (In fact, change the given program so that the head on W' moves in the same way as the head on W, but always printing the symbol 1.)

(2) By (1) it is possible at any time of a computation to erase the content of a work tape (note that the additional work tape used in (1) can be cleared in a trivial way); in particular, one can change a program - without changing the accepted class - such that all work tapes are empty whenever the program stops.

(3) The content of a worktape W can be copied to an empty tape W_1 (using (1), bring the corresponding heads H and H_1 to the rightmost cell scanned by H and copy the content cell by cell).

(4) In our applications the 0-th input tape has the inscription $\underbrace{1 \ldots 1}_{n-\text{times}}$, where n is the cardinality of the structure we consider. One can write the binary representation of n (of length $\leq \log n$) on a work tape. We say "a counter is set to n". Similarly, a counter can be set to n^d for any fixed $d \geq 1$.

Let $\mathcal{L}$ be one of the logics considered in the preceding section and $\mathcal{C}$ the corresponding complexity class. We want to show that for any sentence of $\mathcal{L}$ the class K of its ordered models in in $\mathcal{C}$. We even show that there is a machine M *strongly witnessing* $K \in \mathcal{C}$, that is,

- M accepts K.
- For any $\mathcal{A} \in \mathcal{O}(\tau)$ every run of M, started with $\mathcal{A}$, stops at s_+ or s_-. In particular, if M is deterministic, then M decides K.
- For any $\mathcal{A} \in \mathcal{O}(\tau)$ every run of M satisfies the time or space bounds characteristic for $\mathcal{C}$.

The proof that the class of ordered models of a sentence φ of $\mathcal{L}$ is in $\mathcal{C}$ proceeds by induction on φ. So we have to deal with formulas, too. We therefore introduce

DRAFT

the following notation: For a formula $\varphi(x_1, \ldots, x_l, Y_1, \ldots, Y_r)$ we let

$$\text{ordMod}(\varphi) := \{(\mathcal{A}, a_1, \ldots, a_l, P_1, \ldots, P_r) \mid \mathcal{A} \in \mathcal{O}(\tau), \mathcal{A} \models \varphi[\overline{a}, \overline{P}]\},$$

that is, we consider the ordered models of $\varphi(c_1, \ldots, c_l, P_1, \ldots, P_r)$, a sentence in an enlarged vocabulary.

Theorem 1.4.1 *Let $K \subseteq \mathcal{O}(\tau)$ be a class of ordered structures.*

(a) *If $K \in$ DTC then $K \in$ LOGSPACE.*

(b) *If $K \in$ posTC then $K \in$ NLOGSPACE.*

Proof. By induction on the corresponding formulas φ we show that the class of ordered models of φ is in LOGSPACE and NLOGSPACE, respectively, and that there exists a machine strongly witnessing this fact. We handle both cases simultaneously. By passing to an equivalent formula we can assume that in formulas of FO(posTC) the TC operation does not occur in the scope of a negation symbol.

Suppose that φ is atomic, say for simplicity, $\varphi = Rxy$. We show that there is a machine M strongly witnessing that

$$\{(\mathcal{A}, i, j) \mid \mathcal{A} \in \mathcal{O}(\tau), R^A ij\} \in \text{LOGSPACE}.$$

Let $(\mathcal{A}, i, j) \in \mathcal{O}(\tau \cup \{c, d\})$ be given. Note that the information, whether $R^A ij$ holds, is to be found in the $(i \cdot n + j)$-th square ($n := \|A\|$) of the input tape corresponding to R and (the binary representations) of i and j are available on the input tapes corresponding to c and d. Now it should be clear how a machine, strongly witnessing that ordMod$(Rxy) \in$ LOGSPACE, can be designed.

$\varphi = \neg\psi$: By the remarks above, ψ does not contain TC and, hence, is in FO(DTC). By induction hypthesis, there is a machine M strongly witnessing that ordMod$(\psi) \in$ LOGSPACE. For φ just interchange the roles of s_+ and s_- in M.

$\varphi(x_1, \ldots, x_l) = (\psi \vee \chi)$: By induction hypothesis there are corresponding machines M_ψ and M_χ for $\psi(x_1, \ldots, x_l)$ and $\chi(x_1, \ldots, x_l)$, respectively. Let M be a machine that successively carries out the computations of M_ψ and M_χ accepting the input in case at least one, M_ψ or M_χ, accepts, and rejecting otherwise. (After the computation of M_ψ the work tapes are erased as explained in remark (2) above.)

$\varphi = (\psi \wedge \chi)$: similarly.

$\varphi(x_1, \ldots, x_l) = \exists x\psi$: By induction hypothesis there is a corresponding machine M_0 for $\psi(x_1, \ldots, x_l, x)$. A machine M for φ operates as follows: Suppose M is started with an ordered structure $(\mathcal{A}, a_1, \ldots, a_l)$ where $A = \{0, \ldots, n-1\}$. Then, for $i = 0, \ldots, n-1$, M writes the binary representation of i on a work tape and checks, using M_0, whether $\mathcal{A} \models \psi[a_1, \ldots, a_l, i]$. If the answer is positive at least once, M stops in state s_+, otherwise in s_-.

$\varphi = \forall x\psi$: similarly.

$\varphi = [\mathrm{DTC}_{\overline{x},\overline{y}}\,\psi(\overline{x},\overline{y})]\overline{s}\overline{t}$, where ψ is a formula of FO(DTC): For simplicity, we assume that the free variables of ψ are among $\overline{x},\overline{y}$ and that $\overline{x} = x$, $\overline{y} = y$, $\overline{s} = s$, and $\overline{t} = t$. Choose a machine M_0 strongly witnessing that $\mathrm{ordMod}(\psi) \in$ LOGSPACE. Given $\mathcal{A}$, if there is a φ-path from s to t there is one of length $\leq \|A\|$. Therefore the machine M, we aim at, can be organized as follows: It writes $i := s$ on a work tape and sets a counter to $n\,(:= \|A\|)$, which is used to invoke a subroutine at most n times. M rejects in case the counter becomes negative. Using M_0, the subroutine checks for $j = 0, \ldots, n-1$ whether $\mathcal{A} \models \psi[i,j]$ holds for exactly one j; if not, M rejects. Otherwise, M checks whether j equals t; in the affirmative case M accepts. In the negative case M sets $i := j$.

$\varphi = [\mathrm{TC}_{\overline{x},\overline{y}}\,\psi(\overline{x},\overline{y})]\,\overline{s}\overline{t}$: Once more, for simplicity, we assume that the free variables of ψ are among $\overline{x},\overline{y}$ and that $\overline{x} = x$, $\overline{y} = y$, $\overline{s} = s$, and $\overline{t} = t$. Choose a machine M_0 strongly witnessing $\mathrm{ordMod}(\psi) \in$ NLOGSPACE. We give the basic idea underlying the construction of a machine M for φ: A counter is set to n (the cardinality of the universe), which is used to carry out a subroutine at most n times. M will stop in the rejecting state, when the counter becomes negative. M writes $i := s$ on a work tape. The subroutine starts choosing an element $j \in \{0, \ldots, n-1\}$ nondeterministically (by using a counter to randomly write a $\{0,1\}$ word of length $\log n$ on a tape). Then it checks, using M_0, whether $\psi(i,j)$ holds. If not, M stops in s_-. Otherwise, if $j = t$, M stops in state s_+, and if $j \neq t$ then it sets $i := j$. □

Theorem 1.4.2 *Let $K \subseteq \mathcal{O}(\tau)$ be a class of ordered structures.*

(a) *If $K \in$ IFP then $K \in$ PTIME.*

(b) *If $K \in$ PFP then $K \in$ PSPACE.*

Proof. Again, we give the proof by induction on the formula φ axiomatizing the class K. The cases where φ is $x = y$, $Rx_1 \ldots x_l$, $Xx_1 \ldots x_l$, $\neg\psi$, $(\psi \vee \chi)$, or $\exists x\psi$ are handled as in the preceding proof. The corresponding machines are polynomially time- or space-bounded, if the machines used in the induction hypotheses are. Now, for part (a), suppose that $\varphi = [\mathrm{IFP}_{\overline{x},X}\,\psi(\overline{x},X)]\,\overline{t}$, where X is r-ary, and that M_0 is a machine strongly witnessing that

$$\{(\mathcal{A},\overline{a},R) \mid \mathcal{A} \in \mathcal{O}(\tau), \mathcal{A} \models \psi[\overline{a},R]\} \in \mathrm{PTIME}.$$

The machine M, we look for, contains a subroutine that uses work tapes W and W'. If started with a word of length n^r on W – the code of an r-ary relation R – and empty W', the subroutine writes, invoking the machine M_0, the code of

$$R' := \{\overline{a} \mid \mathcal{A} \models (R\overline{x} \vee \psi)\,[\overline{a},R]\}$$

on the tape W' without changing the content of W.

The machine M for φ operates as follows: It sets $R := \emptyset$ and uses the subroutine to calculate R'. If $R = R'$ it checks whether $R\overline{t}$ or not $R\overline{t}$ and accepts or rejects, respectively. Otherwise it sets $R := R'$, erases the content of W', and again starts the subroutine. Note that $R = R'$ will be achieved after at most n^d calls to the subroutine (by 1.1.1).

For part (b) assume that $\varphi = [\mathrm{PFP}_{\overline{x},X}\,\psi(\overline{x},X)]\,\overline{t}$ with r-ary X and that M_0 is a machine strongly witnessing $\{(\mathcal{A},\overline{a},R) \mid \mathcal{A} \in \mathcal{O}(\tau), \mathcal{A} \models \psi[\overline{a},R]\} \in \mathrm{PSPACE}$.

Given a structure $\mathcal{A}$, the operation F^ψ associated to ψ satisfies $F^\psi_{2^{n^r}-1} = F^\psi_{2^{n^r}}$ (and this set is the fixed-point F^ψ_∞) or $F^\psi_\infty = \emptyset$. The machine M for φ starts its computation on $\mathcal{A}$ by setting a counter to $2^{n^r}-1$ (that is, it writes the word $1\ldots1$ of length n^r on a work tape). Then it proceeds as in the IFP-case, but now using the counter to ensure that the subroutine, which here evaluates

$$R' := \{\overline{a} \mid \mathcal{A} \models \psi[\overline{a},R]\},$$

is invoked at most 2^{n^r} times. When the counter gets zero, it checks whether $R = R'$ and whether Rt. If both questions are answered positively, it accepts, otherwise it rejects.

Theorem 1.4.3 *Let $K \subseteq \mathcal{O}(\tau)$ be a class of ordered structures.*

(a) *If $K \in \Sigma^1_1$ then $K \in$ NPTIME.*

(b) *If $K \in$ SO then $K \in$ PSPACE.*

Proof. (a) Let $K = \mathrm{ordMod}(\varphi)$ where $\varphi = \exists X_1 \ldots \exists X_l \psi$, $\psi \in$ "FO", and the arity of X_i is r_i. By **??**, there is a machine M_0 strongly witnessing that $\mathrm{ordMod}(\psi(X_1,\ldots,X_l))$ is in PTIME (even in LOGSPACE). The machine M for φ, started with $\mathcal{A} \in \mathcal{O}(\tau)$, nondeterministically writes words over {0,1} of length $n^{r_1},\ldots,n^{r_l}$ on different work tapes, which are intended as codes of interpretations $P_1,\ldots,P_l$ of $X_1,\ldots,X_l$. Then it checks, using M_0, whether $\mathcal{A} \models \psi[P_1,\ldots,P_l]$ or not, and stops in an accepting or rejecting state, respectively.

(b) Let $K = \mathrm{ordMod}(\varphi)$ for the formula φ of SO. To gain a machine M witnessing $K \in$ PSPACE we proceed by induction on φ. For φ atomic or of the form $\neg\psi, (\psi \vee \chi)$, or $\exists x\psi$ we argue as in 1.4.1. For $\varphi = \exists X\psi$ with r-ary X the machine M writes the word $1\ldots1$ of length n^r on a work tape W; it then systematically decreases this word, checking in each case with a polynomially space-bounded machine for ψ, whether ψ holds, where X is given by the tape W. □

1.5 The Main Theorem and some consequences

To summarize the result of the preceding section we introduce the following notion

Definition 1.5.1 A logic $\mathcal{L}$ *captures* (or, *characterizes*) a complexity class $\mathcal{C}$, if for all τ with $<\in \tau$ and $K \subseteq \mathcal{O}(t)$ we have,

$$K \in \mathcal{C} \quad \text{iff} \quad K \text{ is axiomatizable in } \mathcal{L}.$$

Theorem 1.5.2 (Main Theorem) (a) FO(DTC) *captures* LOGSPACE.

(b) FO(TC) *captures* NLOGSPACE.

(c) FO(IFP) *captures* PTIME.

(d) Σ^1_1 *captures* NPTIME.

(e) FO(PFP) *captures* PSPACE.

This theorem summarizes the results of the two preceding sections, however, there we have shown that

$$\text{FO(posTC) captures NLOGSPACE.}$$

At the end of this section we will prove that

$$\text{FO(posTC)} \equiv \text{FO(TC) on ordered structures.}$$

Then the proof will be finished.

The descriptive characterizations of complexity classes given by the Main Theorem allow to convert problems, methods, and results in complexity theory into logic and vice versa. In the remainder of this section we give some illustrating examples, where we first stay with ordered structures and subsequently discuss the role of order.

Sentences φ and ψ in a vocabulary τ with $<\in \tau$ are said to be *equivalent in ordered structures*, if for all ordered τ-structures $\mathcal{A}$,

$$\mathcal{A} \models \varphi \quad \text{iff} \quad \mathcal{A} \models \psi.$$

Corollary 1.5.3 *In ordered structures every* FO(IFP) *sentence is equivalent to a* FO(IFP) *sentence in which* IFP *occurs at most once. The same applies to* FO(PFP) *and* PFP*.*

Proof. Suppose $\varphi \in \text{FO(IFP)}[\tau]$ with $<\in \tau$. Then $\text{ordMod}(\varphi) \in \text{PTIME}$. Now the claim follows from 1.3.5. □

Corollary 1.5.4 *Let $\mathcal{C}$ be one of the complexity classes mentioned in the Main Theorem. If $K \in \mathcal{C}$ then there is a Turing machine M strongly witnessing $K \in \mathcal{C}$, that is,*

- *M accepts K,*

Draft

- *every run stops in the accepting or in the rejecting state,*
- *every run fulfills the time or space bounds characteristic for* $\mathcal{C}$.

Proof. Let $\mathcal{L}$ be the logic capturing $\mathcal{C}$. Then there is a sentence of $\mathcal{L}$ axiomatizing K. By the results of section 1.4, we know that for every class K axiomatizable in $\mathcal{L}$ there is a machine strongly witnessing $K \in \mathcal{C}$. □

An immediate consequence of the Main Theorem is the equivalence of (i) and (ii):

(i) PTIME = PSPACE

(ii) FO(IFP) ≡ FO(PFP) on ordered structures.

Note, however, that here PTIME and PSPACE are defined using classes of structures and not languages over alphabets. Does (i) mean the same as PTIME = PSPACE in complexity theory? We want to show this by making clear that here and in complexity theory we deal only with different presentations of a complexity class.

If $\mathcal{C}$ is a complexity class of complexity theory, we denote by $\mathcal{C}'$ the corresponding complexity class of structures. For example, for PTIME we have

> $\mathcal{C}$ consists of all languages L, $L \subseteq \mathbb{A}^+$ for some alphabet $\mathbb{A}$, such that there exists a Turing machine accepting L in polynomial time

and

> $\mathcal{C}'$ consists of all classes K, $K \subseteq \mathcal{O}(\tau)$ for some τ with $<\in \tau$, such that there is a Turing machine M accepting K in polynomial time.

In the following let $\mathcal{C}, \mathcal{C}_1, \mathcal{C}_2$ range over the complexity classes LOGSPACE, NLOGSPACE, PTIME, NPTIME and PSPACE of complexity theory.

Our first consideration will show that $\mathcal{C} \subseteq \mathcal{C}'$ up to a natural transition from words to ordered structures, thereby using word models (already introduced in section ??). Fix an alphabet $\mathbb{A}$ and let $\tau(\mathbb{A})$ be the vocabulary $\{<\} \cup \{P_a \mid a \in \mathbb{A}\}$ with unary P_a. If $u \in \mathbb{A}^+$, denote by K_u the class of structures of the form

$$(B, <, (P_a)_{a \in \mathbb{A}}),$$

where the cardinality of B equals the length of u, $<$ is an ordering of B, and P_a corresponds to the positions in u carrying an a.

For $L \subseteq \mathbb{A}^+$ set

$$K(L) := \bigcup_{u \in L} K_u.$$

Clearly, $K(L) \subseteq \mathcal{O}(\tau(\mathbb{A}))$ and, since

$$K(\mathbb{A}^+) = \mathrm{ordMod}(\forall x(\bigvee_{a \in \mathbb{A}} (P_a x \wedge \bigwedge_{\substack{b \in \mathbb{A} \\ b \neq a}} \neg P_b x))), \tag{1}$$

Draft

we have $K(\mathbb{A}^+) \in$ LOGSPACE.

One can easily show (see Exercise ?? below) that

(2) $$\text{for } L \subseteq \mathbb{A}^+, \quad L \in \mathcal{C} \text{ iff } K(L) \in \mathcal{C}',$$

thus obtaining "$\mathcal{C} \subseteq \mathcal{C}'$ up to transitions".

Now we turn to a statement showing $\mathcal{C}' \subseteq \mathcal{C}$ up to a transition from ordered structures to words. Let τ with $<\in \tau$ be given. Set $\mathbb{A}_0 := \{\alpha, 0, 1, \omega\}$. For $\mathcal{A} \in \mathcal{O}(\tau)$ let $u_\mathcal{A}$ be the word in $\mathbb{A}_0^+$ obtained by concatenating the inscriptions on all input tapes of a Turing machine started with input $\mathcal{A}$ (where we include the "virtual letters" α and ω). For a class $K \subseteq \mathcal{O}(\tau)$ set

$$L(K) := \{u_\mathcal{A} \mid \mathcal{A} \in K\}.$$

Clearly, for given τ there is a polynomial $p \in \mathbb{N}[x]$ such that for all $\mathcal{A} \in \mathcal{O}(\tau)$ we have $\|A\| \leq |u_\mathcal{A}| \leq p(\|A\|)$. In particular, for $p(x) := x^d$ we have $\log \|A\| \leq \log |u_\mathcal{A}| \leq d \log \|A\|$. Invoking these relations one easily shows

(3) $$L(\mathcal{O}(\tau)) \in \text{LOGSPACE}$$

(4) $$\text{for } K \subseteq \mathcal{O}(\tau), \quad K \in \mathcal{C} \text{ iff } L(K) \in \mathcal{C}.$$

From (2) and (4) we infer

Proposition 1.5.5 $\mathcal{C}_1 \subseteq \mathcal{C}_2$ *iff* $\mathcal{C}_1' \subseteq \mathcal{C}_2'$.

Proof. First, suppose $\mathcal{C}_1 \subseteq \mathcal{C}_2$ and let $K \in \mathcal{C}_1'$ where $K \subseteq \mathcal{O}(\tau)$. Then by(4), $L(K) \in \mathcal{C}_1$ and, by hypothesis, $L(K) \in \mathcal{C}_2$. Therefore, $K \in \mathcal{C}_2$ (again by (4)). Now assume $\mathcal{C}_1' \subseteq \mathcal{C}_2'$ and let $L \in \mathcal{C}_1$. Then $K(L) \in \mathcal{C}_1'$ by (2), and hence, $K(L) \in \mathcal{C}_2'$. Therefore $L \in \mathcal{C}_2$ by (2). □

Together with the Main Theorem we have

Corollary 1.5.6 (a) FO(IFP) $\equiv$ FO(PFP) *on ordered structures*
iff PTIME = PSPACE (*in complexity theory*).

(b) FO(IFP) $\equiv \Sigma_1^1$ *on ordered structures*
iff PTIME = NPTIME (*in complexity theory*).

□

Moreover the preceding argumentation shows that in order to get FO(IFP) $\equiv$ FO(PFP) on ordered structures (or, equivalently, PTIME = PSPACE) it suffices to prove that FO(IFP) $\equiv$ FO(PFP) holds for ordered τ-structures, where τ besides $<$ only contains unary relation symbolds. Since one can restrict to languages over $\{0, 1\}$ even a single unary relation symbol suffices.

Whereas the preceding corollary contains the translation of *problems* from complexity theory to logics we now turn to the translation of a *result*. In complexity theory one shows

$$\text{LOGSPACE} \subseteq \text{NLOGSPACE} \subseteq \text{PTIME} \subseteq \text{NPTIME} \subseteq \text{PSPACE}$$

Draft

and

$$\text{LOGSPACE} \neq \text{PSPACE}.$$

Hence, by the Main Theorem,

Corollary 1.5.7 *On ordered structures,*

(a) $\text{FO(DTC)} \leq \text{FO(TC)} \leq \text{FO(IFP)} \leq \Sigma^1_1 \leq \text{FO(PFP)}$

(b) $\text{FO(DTC)} \not\equiv \text{FO(PFP)}$.

□

Note that most of the $\leq$-relations in (a) are immediate; the remaining ones and part (b) will be obtained in Chapter ?? by purely modeltheoretic means.

For any of the complexity classes $\mathcal{C}$ introduced so far, in complexity theory one defines the class co-$\mathcal{C}$ to be the class of complements of languages in $\mathcal{C}$, that is, for any alphabet $\mathbb{A}$ and $L \subseteq \mathbb{A}^+$,

$$L \in \text{co-}\mathcal{C} \quad \text{iff} \quad (\mathbb{A}^+ \setminus L) \in \mathcal{C}.$$

Clearly, any determinstic class $\mathcal{C}$ is closed under complements, that is, $\mathcal{C} = \text{co-}\mathcal{C}$. Similarly we define the class co-$\mathcal{C}'$ as the class of complements of classes of structures in $\mathcal{C}'$; more precisely, for τ with $<\in \tau$ and $K \subseteq \mathcal{O}(\tau)$, we set

$$K \in \text{co-}\mathcal{C}' \quad \text{iff} \quad (\text{O}(\tau) \setminus K) \in \mathcal{C}'.$$

Exercise 1.5.8 Show that Proposition 1.5.5 remains true, if one allows $\mathcal{C}_1$ and $\mathcal{C}_2$ to range over the complexity classes mentioned in the Main Theorem and their complements. (Use (1)-(4) before 1.5.5.)

We now are going to discuss the role of order and get information whether and to what extent orderings can be avoided. Let $\mathcal{A}$ be a structure not necessarily ordered. We already mentioned that in order to consider $\mathcal{A}$ as an input for a Turing machine, we have to represent it as a string (or a sequence of strings), for example, by labeling the elements of $\mathcal{A}$ in some way. Taking, say, the lexicographic ordering of the labels, we get an ordering on A and, hence, an ordered structure. Now, if $\mathcal{A}$, say, is a graph we can state questions such as "Is there a path from the 5-th to the 28-th element?" whose answer depends on this ordering and is senseless for $\mathcal{A}$ itself. The following framework enables us to concentrate on questions intrinsic to $\mathcal{A}$.

Given a class K of (unordered) τ-structures, we enlarge τ to $\tau_< := \tau \cup \{<\}$ and pass to the class $K_<$ of $\tau-<$-structures,

$$K_< := \{(\mathcal{A}, <) \mid \mathcal{A} \in K, < \text{ an ordering of } A\}.$$

Draft

If $\mathcal{L}$ is a logic capturing the complexity class $\mathcal{C}$ we have

$$K_< \in \mathcal{C} \quad \text{iff} \quad \text{there is } \varphi \in \mathcal{L}[\tau_<] \text{ such that } K_< = \text{ordMod}(\varphi).$$

The sentence φ on the rigth side of the equivalence is order-invariant, since for any $\mathcal{A}$ and orderings $<_1$ and $<_2$ of A we have

$$(\mathcal{A}, <_1) \in K_< \text{ iff } (\mathcal{A}, <_2) \in K_<,$$

and thus

$$(\mathcal{A}, <_1) \models \varphi \text{ iff } (\mathcal{A}, <_2) \models \varphi.$$

Now, if every order-invariant sentence $\varphi \in \mathcal{L}[\tau_<]$ is equivalent to a sentence $\psi \in \mathcal{L}[\tau]$, we have

$$K_< \in \mathcal{C} \text{ iff } \text{there is } \psi \in \mathcal{L}[\tau] \text{ such that } K_< = \text{ordMod}^{\tau <}(\psi)^7$$

i.e.

$$K_< \in \mathcal{C} \text{ iff } \text{there is } \psi \in \mathcal{L}[\tau] \text{ such that } K_< = \text{Mod}^{\tau}(\psi).$$

In general this does not hold. (For first-order logic we gave a counterexample already in ??.) To give a counterexample for FO(DTC), let $K = \text{EVEN}(\tau)$ with $\tau = \emptyset$ be the class of sets of even cardinality. Since $K_< \in$ LOGSPACE, there is a sentence φ of FO(DTC) such that

$$K_< = \text{ordMod}(\varphi);$$

but we shall see in ?? that for no sentence ψ of FO(DTC) - even of FO(PFP) -

$$K = \text{ordMod}(\psi).$$

These observations lead us to a stronger notion of what it means to capture a complexity class.

Definition 1.5.9 Let $\mathcal{L}$ be a logic and $\mathcal{C}$ a complexity class. $\mathcal{L}$ *strongly captures* $\mathcal{C}$ (or, *stronly characterizes* $\mathcal{C}$, if for all vocabularies τ and all classes K of τ-structures,

$$K_< \in \mathcal{C} \text{ iff } K \text{ is axiomatizable in } \mathcal{L}.$$

The proof of the following proposition is left to the reader as an exercise.

Proposition 1.5.10 *If $\mathcal{L}$ strongly captures $\mathcal{C}$ then $\mathcal{L}$ captures $\mathcal{C}$.*

[7] We use $\text{Mod}^{\sigma}(\ldots)$ with upper index σ to rule out any ambiguity on the vocabulary considered.

Draft

The converse is false: The counterexample given before the definition shows that FO(PFP) does not capture PSPACE strongly. For the class EVEN(τ) we used as a counterexample, we have EVEN(τ)$_<$ $\in$ LOGSPACE, and EVEN(τ) is not axiomatizable in FO(PFP). Since FO(DTC) $\leq$ FO(TC) $\leq$ FO(IFP) $\leq$ FO(PFP) holds for arbitrary structures, we see that none of these logics strongly characterizes the corresponding complexity class. The result cannot be extended to Σ^1_1 and NPTIME, since we know $\Sigma^1_1 <$PFP only on *ordered structures*. In fact, we have

Theorem 1.5.11 Σ^1_1 *strongly captures* NPTIME.

Proof. Let τ be arbitrary and K be a class of τ-structures. If $K = Mod^\tau(\varphi)$ for some $\Sigma^1_1[\tau]$ sentence φ, then $K_< = \mathrm{ordMod}^{\tau<}(\varphi)$, hence $K_< \in$ NPTIME by the Main Theorem. Conversely, if $K_< \in$ NPTIME, then, again by the Main Theorem, there is a sentence $\varphi \in \Sigma^1_1[\tau_<]$, say $\varphi = \exists X_1 \ldots \exists X_m \psi$ with $\psi \in$ "FO", such that $K_< = \mathrm{ordMod}^{\tau<}(\varphi)$. Set

$$\chi \quad := \quad \exists < \exists X_1 \ldots \exists X_m (\text{“} < \text{ is an ordering}'' \wedge \psi).$$

Then $\chi \in \Sigma^1_1[\tau]$, and for any τ-structure $\mathcal{A}$ we have

$$\begin{array}{lll} \mathcal{A} \models \chi & \text{iff} & \text{there is an ordering } <^A \text{ on } A \text{ with } (\mathcal{A}, <^A) \models \varphi \\ & \text{iff} & \text{there is an ordering } <^A \text{ on } A \text{ with } (\mathcal{A}, <^A) \in K_< \\ & \text{iff} & \mathcal{A} \in K, \end{array}$$

that is, $K = Mod^\tau(\chi)$. □

Corollary 1.5.12 *The following are equivalent:*

(i) NPTIME = PTIME

(ii) SO $\equiv$ FO(IFP).

Proof. If (ii) holds, then $\Sigma^1_1 \leq$ FO(IFP) on ordered structures, thus NPTIME $\leq$ PTIME. Conversely, if NPTIME $\equiv$ PTIME then, on ordered structures, $\Sigma^1_1 \equiv$ FO(IFP). As Σ^1_1 is closed under existential quantifications and FO(IFP) under negations and dijunctions, an easy induction yields SO $\equiv$ FO(IFP). □

Theorem 1.5.13 Π^1_1 *strongly captures* co-NPTIME.

Proof. Let τ be arbitrary and K be a class of τ-structures. If Str(τ) denotes the class of all τ -structures, then

$$(+) \qquad (\mathrm{Str}(\tau) \setminus K)_< = \mathcal{O}(\tau) \setminus K_<.$$

draft

Therefore,

$$\begin{array}{lll} K \in \Pi^1_1 & \text{iff} & \text{there is } \psi(X_1,\dots,X_m) \in \text{“FO}[\tau]\text{”} \text{ such that} \\ & & K = \mathrm{Mod}(\forall X_1 \dots \forall X_m \psi) \\ & \text{iff} & \text{there is } \chi(X_1,\dots,X_m) \in \text{“FO}[\tau]\text{”} \text{ such that} \\ & & \mathrm{Str}(\tau) \setminus K = \mathrm{Mod}(\exists X_1 \dots \exists X_m \chi) \\ & \text{iff} & (\mathrm{Str}(\tau) \setminus K)_< \in \text{NPTIME} \qquad \text{(by 1.5.11)} \\ & \text{iff} & (\mathcal{O}(\tau) \setminus K_<) \in \text{NPTIME} \qquad \text{(by (+)} \\ & \text{iff} & K_< \in \text{co-NPTIME}. \end{array}$$

□

Corollary 1.5.14 NPTIME $\equiv$ co-NPTIME *iff* $\Sigma^1_1 \equiv \Pi^1_1$.

Proof. As an example we show the implication from left to right. Let K be a class of structures. Then

$$\begin{array}{lll} K \in \Sigma^1_1 & \text{iff} & K_< \in \text{NPTIME} \\ & \text{iff} & K_< \in \text{co-NPTIME} \quad \text{(by hypothesis)} \\ & \text{iff} & K \in \Pi^1_1. \end{array}$$

□

Corollary 1.5.15 NPTIME =co-NPTIME *iff* SO $\equiv \Sigma^1_1$.

Proof. By the preceding corollary it suffices to prove

$$\Pi^1_1 \equiv \Sigma^1_1 \quad \text{iff} \quad \text{SO} \equiv \Sigma^1_1.$$

For a logic $\mathcal{L}$ we write $\varphi \underset{\sim}{\in} \mathcal{L}$ to express that φ is equivalent to an $\mathcal{L}$-sentence. Clearly,

$$\begin{array}{lll} \varphi \in \Sigma^1_1 & \text{implies} & \neg\varphi \underset{\sim}{\in} \Pi^1_1, \\ \varphi \in \Pi^1_1 & \text{implies} & \neg\varphi \underset{\sim}{\in} \Sigma^1_1. \end{array}$$

Now suppose that SO $\equiv \Sigma^1_1$ and therefore, $\Pi^1_1 \leq \Sigma^1_1$. Let $\varphi \in \Sigma^1_1$. Then $\neg\varphi \underset{\sim}{\in} \Pi^1_1$ and hence, $\neg\varphi \underset{\sim}{\in} \Sigma^1_1$. Therefore, $\varphi \underset{\sim}{\in} \Pi^1_1$. Conversely, assume that $\Pi^1_1 \equiv \Sigma^1_1$. One easily shows that the class Σ^1_1 is closed – up to equivalence – under $\vee$ and existential first- and second-order quantification. For closure under $\neg$ argue as follows: Suppose $\varphi = \neg\psi$ and $\psi \in \Sigma^1_1$. Then $\neg\psi \underset{\sim}{\in} \Pi^1_1 \equiv \Sigma^1_1$. □

Exercise 1.5.16 Define the logic SO(PFP) and show that it captures PSPACE strongly. In **??** we shall show by a purely modeltheoretic proof that, on ordered structures, SO(PFP) $\equiv$ FO(PFP).

Draft

The question whether the classes LOGSPACE, NLOGSPACE, and PTIME can be strongly captured by a logic is a major issue of descriptive complexity theory. Of course, one has to give a precise definition of what is meant by a "logic" in this context. Moreover, to rule out trivial positive answers, one has to require some conditions of effectiveness. Note that in the capturing results obtained in the preceding sections, we effectively associated to every sentence a Turing machine accepting its models. And to every Turing machine accepting a class of models we effectively associated a sentence axiomatizing this class. Up to now all attempts to find a logic that strongly captures PTIME have failed. In ?? we shall come back to such attempts when discussing so-called generalized quantifiers.

We close this section by a result, which completes the proof of the Main Theorem.

Theorem 1.5.17 *On ordered structures,* FO(posTC) ≡ FO(TC).

Proof. We make use of the fact (shown in??) that, on ordered structures, every formula of FO(posTC) is equivalent to a formula of the form

$$[\mathrm{TC}_{\overline{x},\overline{y}}\,\psi(\overline{x},\overline{y})]\,\widetilde{\mathrm{min}}\widetilde{\mathrm{max}}$$

with first-order ψ.

The proof of the theorem proceeds by induction on FO(TC) formulas, the only nontrivial case being the negation step. By the induction hypothesis and the fact just mentioned we may assume that

$$\varphi = \neg\,[\mathrm{TC}_{\overline{x},\overline{y}}\,\psi(\overline{x},\overline{y})]\,\overline{s}\overline{t}$$

with first-order ψ. By the Main Theorem (more precisely, by 1.4.1) there is a Turing machine M_0 strongly witnessing that ordMod(ψ) ∈ LOGSPACE.

Suppose $\overline{x} = x_1 \ldots x_r$. Given a structure $\mathcal{A}$ and $\overline{a}, \overline{b} \in A^r$, let $d_\psi(\overline{a},\overline{b})$, be the length of the shortest ψ-path connecting $\overline{a}$ and $\overline{b}$,

$$d_\psi(\overline{a},\overline{b}) \;:=\; \min\{k>0 \mid \text{there exist } \overline{a}_0=\overline{a},\overline{a}_1,\ldots,\overline{a}_k=\overline{b} \text{ such that } \mathcal{A}\models\psi[\overline{a}_i,\overline{a}_{i+1}] \text{ for } i<k\},$$

where $d_\psi(\overline{a},\overline{b}) := \infty$ in case the set on the right side is empty. Note that

- if $d_\psi(\overline{a},\overline{b}) < \infty$ then $0 < d_\psi(\overline{a},\overline{b}) \leq \|A\|^r$
- $d_\psi(\overline{a},\overline{c}) \leq d_\psi(\overline{a},\overline{b}) + d_\psi(\overline{b},\overline{c})$.

Moreover, $\neg\,[\mathrm{TC}_{\overline{x},\overline{y}}\,\psi(\overline{x},\overline{y})]\,\overline{s}\overline{t}$ is equivalent to

$$\text{“}\|\{\overline{v} \mid d_\psi(\overline{s},\overline{v})<\infty\}\| = \|\{\overline{v} \mid d_{(\psi(\overline{x},\overline{y})\,\wedge\,\neg\overline{y}=\overline{t})}(\overline{s},\overline{v})<\infty\}\|\text{”}.$$

We first show that

there is a nondeterministic log space-bounded machine M such that
for any ordered structure $(\mathcal{A}, \overline{a}, \overline{l}, \overline{w}, \overline{w}')$,
if $\|\{\overline{b} \mid d_\psi(\overline{a}, \overline{b}) \leq \overline{l}\}\| = \overline{w}$ then

$$M \text{ accepts } (\mathcal{A}, \overline{a}, \overline{l}, \overline{w}, \overline{w}') \text{ iff } \|\{\overline{b} \mid d_\psi(\overline{a}, \overline{b}) \leq \overline{l}+1\}\| = \overline{w}',$$

where the natural numbers $\leq \|A\|^r$ are given by their $\|A\|$-adic representations $\overline{l} = l_0 \dots l_r$, $\overline{w} = w_0 \dots w_r$, $\overline{w}' = w'_0 \dots w'_r$.

We present the machine M. When during its computation M checks, whether $\mathcal{A} \models \psi[\overline{c}, \overline{d}]$ holds or not, this is done by invoking M_0. Started with $(\mathcal{A}, \overline{a}, \overline{l}, \overline{w}, \overline{w}')$, M first sets a counter to $\overline{w}'$. Then, for every $\overline{b} \in A^r$, it carries out either (1) or (2), the choice being done nondeterministically:

(1) M nondeterministically guesses a path witnessing $d_\psi(\overline{a}, \overline{b}) \leq \overline{l}+1$ and decreases the counter by one; in case the counter is zero it rejects.

(2) Using an additional counter with the initial value $\overline{w}$ nondeterministically guesses $\overline{w}$ many tuples $\overline{c} \in A^r$ together with a proof that $d_\psi(\overline{a}, \overline{c}) \leq \overline{l}$; for each such $\overline{c}$ it shows that $\neg\psi[\overline{c}, \overline{b}]$ (thus finally proving that $d_\psi(\overline{a}, \overline{b}) > \overline{l}+1$). In case $\overline{l} = 0$ it shows $\neg\psi[\overline{a}, \overline{b}]$.

If, finally, all $\overline{b} \in A^r$ have been dealt with and the counter is 0, M accepts.

Since M is log space-bounded there is, by the Main Theorem, a formula $\chi_\psi(\overline{u}, \overline{y}, \overline{z}, \overline{z}') \in$ FO(posTC) axiomatizing the class accepted by M. Then, noting that $1\underbrace{0 \dots 0}_{\text{r times}}$ is (the representation of) $\|A\|^r$, we get a formula $\rho_\psi(\overline{u}, \overline{z}) \in$ FO(posTC) with the meaning

$$\text{“}\|\{\overline{v} \mid d_\psi(\overline{u}, \overline{v}) < \infty\}\| = \overline{z}\text{”}$$

or, equivalently, with the meaning

$$\text{“}\|\{\overline{v} \mid d_\psi(\overline{u}, \overline{v}) \leq \|A\|^r\}\| = \overline{z}\text{”}$$

by setting

$$\rho_\psi(\overline{u}, \overline{z}) \quad := \quad [\mathrm{TC}_{\overline{y}\overline{z}, \overline{y}'\overline{z}'}(\chi_\psi(\overline{u}, \overline{y}, \overline{z}, \overline{z}') \wedge \overline{y}' = \overline{y}+1)]\, \widetilde{\min}\widetilde{\min} 10 \dots 0\overline{z}.$$

Since we saw above that $\varphi = \neg\,[\mathrm{TC}_{\overline{x}, \overline{y}}\, \psi(\overline{x}, \overline{y})]\, \overline{s}\overline{t}$ is equivalent to

$$\text{“}\|\{\overline{v} \mid d_\psi(\overline{s}, \overline{v}) < \infty\}\| = \|\{\overline{v} \mid d_{(\psi(\overline{x}, \overline{y}) \wedge \neg\, \overline{y} = \overline{t})}(\overline{s}, \overline{v}) < \infty\}\|\text{”},$$

we obtain that φ is equivalent to the formula

$$\exists \overline{z}\, (\rho_\psi(\overline{s}, \overline{z}) \wedge \rho_{(\psi \wedge \neg\, \overline{y} = \overline{t})}(\overline{s}, \overline{z})),$$

a formula of FO(posTC). □

As a consequence of part (b) of the Main Theorem whose proof we just have completed we get, since FO(TC) is closed under negation:

Corollary 1.5.18 NLOGSPACE =co-NLOGSPACE. □

In the preceding considerations, when treating questions of effectiveness, we have restricted ourselves to ordered structures $\mathcal{A}$, as they offer a natural way of encoding by passing to the unique isomorphic copy whose universe is $\{0, \ldots, \|A\| - 1\}$ and whose order is the natural order on $\{0, \ldots, \|A\| - 1\}$. There is an alternative way, namely the direct restriction to structures whose universe forms an initial segment of the natural numbers. The following exercise shows that both frameworks are equivalent.

Exercise 1.5.19 Let τ be an arbitrary vocabulary. A *numerical* τ-structure is a τ-structure whose universe is an initial segment of the natural numbers, that is, the universe is the set $\{0, \ldots, n\}$ for some n. For a class K of τ-structures let

$$K_{\text{num}} \quad := \quad \{\mathcal{A} \in K \mid \mathcal{A} \text{ is numerical}\}$$

be the set of numerical structures in K, and for a sentence φ let

$$\text{Mod}_{\text{num}}(\varphi) \quad := \quad \{\mathcal{A} \mid \mathcal{A} \models \varphi, \mathcal{A} \text{ numerical}\}$$

be the set of its numerical models.

Let $\mathcal{C}$ and $\mathcal{L}$ be any of the complexity classes and logics, respectively, considered so far. Show

(a) For any class K : $K_< \in \mathcal{C}$ iff $K_{\text{num}} \in \mathcal{C}$ (that is, there is an algorithm according to $\mathcal{C}$ that decides for any numerical structure $\mathcal{A}$ whether $\mathcal{A} \in K_{\text{num}}$).

(b) $\mathcal{L}$ strongly captures $\mathcal{C}$ iff for any τ and any class K of τ-structures, $K_{\text{num}} \in \mathcal{C}$ iff there is a sentence φ of $\mathcal{L}[\tau]$ such that $K_{\text{num}} = Mod_{\text{num}}(\varphi)$.

(c) $\mathcal{L}$ captures $\mathcal{C}$ iff for any τ and any class K of τ-structures, $K_{\text{num}} \in \varphi$ iff there is an $\mathcal{L}$ sentence φ of $\mathcal{L}[\tau \cup \{<\}]$ such that $K_{\text{num}} = \{\mathcal{A} \mid (\mathcal{A}, <^A) \models \varphi, A = \{0, \ldots, n\}$ for some n and $<^A$ is the natural ordering on $A\}$.

Draft