Descriptive Complexity Theory

1.-2.Vorlesungen

Introduction

To illustrate the idea of DCT (Descriptive Complexity Theory), consider a finite structure which presents a list of cities and flights between them by various airlines. So this will look like $\mathcal{S} = (C, A, E)$ where C is the set of cities, A is the set of airlines and $E(a, x, y)$ means that airline a flies from city x to city y.

Now we can consider various relations over this structure $\mathcal{S}$: Is there a flight from Vienna to Kabul with Aeroflot?

$$E(\text{Aeroflot,Vienna,Kabul})$$

Is there such a flight with one stopover?

$$\exists x(E(\text{Aeroflot,Vienna}, x) \text{ and } E(\text{Aeroflot}, x, \text{Kabul}))$$

Is there any way of flying from Vienna to Kabul at all, with any number of stops on any of the listed airlines?

$$\bigvee_{1\leq i\leq n} \exists x_1, \dots, x_i \exists a_1, \dots, a_{i-1}(x_1 = \text{ Vienna and } E(a_1, x_1, x_2) \text{ and } \cdots \\ \text{and } E(a_{i-1}, x_{i-1}, x_i) \text{ and } x_i = \text{ Kabul}),$$

where n is the number of cities (minus 2 if you like). Clearly the latter, unlike the first two examples, is not in first-order logic, because the length of the formula depends on the number of cities. We'll introduce some natural logics making use of fixed-point operators (and second-order quantífiers) which go beyond first-order logic and can express natural relations on finite structures which are not expressible in first-order logic.

It is not difficult to see that finite structures can be treated as inputs to a Turing machine and therefore we can ask about the computational complexity of relations such as those above. It will turn out that there is a close connection between the computational complexity of a relation and its definability in a certain logic, and this connection is the main theme of descriptive complexity theory.

If M is a finite nonempty set then $\mathcal{P}(M)$ denotes the power set of M. A function $F : \mathcal{P}(M) \to \mathcal{P}(M)$ induces a sequence $\emptyset, F(\emptyset), F(F(\emptyset)), \cdots$ of subsets of M. For ease of notation we write $F_0 = \emptyset$, $F_{n+1} = F(F_n)$. If there is an n_0 such that $F(F_{n_0}) = F_{n_0}$ then of course $F_n = F_{n_0}$ for all $n \geq n_0$ and we write F_∞ for F_{n_0}. We say that the *fixed point F_∞ of F exists.* If F_∞ does not exist then by convention we write $F_\infty = \emptyset$.

One case in which F_∞ must exist is if F is *inflationary*; this means that $X \subseteq F(X)$ for all X. The following is rather obvious.

Lemma 1 *(a) For any F there are $m < 2^{|M|}$ and $l > 0$ such that $F_k = F_{k+l}$ for all $k \geq m$.*
(b) If F_∞ exists then $F_\infty = F_{2^{|M|}-1}$.
(c) If F is inflationary then F_∞ exists and $F_\infty = F_{|M|}$.

Now we consider operations F as above which result from the interpretation of a formula. Fix a finite relational vocabulary τ. So τ specifies a finite set of constant symbols and a finite set of relation symbols, of various arities. Let $\varphi(x_1, \ldots, x_k, \bar{u}, X, \bar{Y})$ be a formula in the second-order language where $\bar{u}$ is a sequence of (first-order) variables, X is a relation variable of arity k and $\bar{Y}$ is a sequence of relation variables. Also fix a finite structure $\mathcal{A}$ which interprets $\bar{u}$ as some tuple $\bar{b}$ of elements of its universe A and $\bar{Y}$ as some tuple $\bar{S}$ of relations (of various arities) over its universe. Then we can use $\varphi, \mathcal{A}, \bar{b}$ and $\bar{S}$ to define an operation, which we write just as $F^\varphi : \mathcal{P}(A^k) \to \mathcal{P}(A^k)$, as follows:

$$F^\varphi(R) = \{(a_1, \ldots, a_k) \mid \mathcal{A} \vDash \varphi(a_1, \ldots, a_k, \bar{b}, R, \bar{S})\}.$$

We set F^φ_∞ to be the fixed point of F^φ_n if it exists, $\emptyset$ otherwise.

Example (path-connectedness in a graph). Suppose that $\mathcal{G} = (G, E^G)$ is a graph, i.e., an irreflexive, symmetric binary relation. Consider the formula

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

Then $F^\varphi_n = \{(a, b) \mid \text{There is a path of length} \leq n \text{ from } a \text{ to } b\}$. so the fixed-point F^φ_∞ exists and equals $\{(a, b) \mid \text{There is a path from } a \text{ to } b\}$. Although

F^{φ} is not necessarily inflationary, we can make it inflationary by considering instead $\varphi^* = \varphi \vee Xxy$.

FO(PFP) and FO(IFP)

By adding fixed-point operators to first-order logic we obtain the above logics, defined as follows. Again fix a finite relational vocabulary τ. The class of formulas of FO(PFP) is given inductively as follows:

• Any atomic *second-order* formula is a formula.

• If φ, ψ are formulas then so are $\sim \varphi$, $(\varphi \vee \psi)$ and $\exists x \varphi$ for any first-order variable x.

• If φ is a formula, $\bar{x}$ is a sequence of first-order variables, X is a second-order variable and $\bar{t}$ is a sequence of first-order terms (i.e., first-order variables or constant symbols) such that length$(\bar{x})$ = length$(\bar{t})$ = arity(X) then $[\mathrm{PFP}_{\bar{x},X}\varphi]\bar{t}$ is a formula.

Free (first-order and second-order) variables are defined in the natural way; for example the free variables of $[\mathrm{PFP}_{\bar{x},X}\varphi]\bar{t}$ are those of $\bar{t}$ together with those of φ other than X and the components of $\bar{x}$.

The semantics is as follows: Suppose that the relation variable X has arity k and the free variables of $[\mathrm{PFP}_{\bar{x},X}\varphi]\bar{t}$ are among $\bar{u}$ (first-order variables), $\bar{Y}$ (relation variables). If $\bar{b}$, $\bar{S}$ are interpretations of $\bar{u}$, $\bar{Y}$ in $\mathcal{A}$ then

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

The above is "partial fixed-point logic". To obtain "inflationary fixed-point logic" we replace PFP by IFP in the inductive definition of formula and in the semantics replace F^{φ}_{∞} by $F^{X\bar{x}\vee\varphi}_{\infty}$ (thereby insuring that our operator is inflationary).

Example (path-connectedness again). In the language of graphs, the formula

$$\varphi(x, y) = [\mathrm{IFP}_{xy,X}(Exy \vee \exists z(Xxz \vee Ezy))]xy$$

of FO(IFP) expresses that x, y are connected by a path. Thus the class of connected graphs is axiomatisable in FO(IFP); it is not axiomatisable in first-order logic (using Ehrenfeucht-Fraissé games).

Example(even cardinality). In the language $\{<, S, \min, \max\}$ the sentence

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

of FO(IFP) together with the linear ordering axioms axiomatises the class of linear orderings of even cardinality. Again this is not expressible in first-order logic.

FO(TC) and FO(DTC)

These logics are potentially weaker than FO(IFP) and FO(PFP). Again fix a finite relational vocabulary τ. The formulas of FO(TC) are defined inductively as follows.

• Any atomic *first-order* formula is a formula.

• If φ, ψ are formulas then so are $\sim \varphi$, $(\varphi \vee \psi)$ and $\exists x \varphi$ for any first-order variable x.

• If φ is a formula then so is $[\mathrm{TC}_{\bar{x},\bar{y}}\varphi]\bar{s}\bar{t}$, provided the variables in $\bar{x}, \bar{y}$ are all distinct, $\bar{s}, \bar{t}$ are sequences of terms (first-order variables or constant symbols) and the lengths of $\bar{x}, \bar{y}, \bar{s}, \bar{t}$ are all the same.

As expected the free variables of $[\mathrm{TC}_{\bar{x},\bar{y}}\varphi]\bar{s}\bar{t}$ are those of φ without the variables in $\bar{x}, \bar{y}$, together with the variables in $\bar{s}, \bar{t}$. The semantics is as follows: If the free variables in $[\mathrm{TC}_{\bar{x},\bar{y}}\varphi]\bar{s}\bar{t}$ are among $\bar{u}$ and $\bar{b}$ is an interpretation in the structure $\mathcal{A}$ of $\bar{u}$, then

$$\mathcal{A} \vDash [\mathrm{TC}_{\bar{x},\bar{y}}\varphi]\bar{s}\bar{t}\ [\bar{b}] \text{ iff } (\bar{s}[\bar{b}], \bar{t}[\bar{b}]) \in \mathrm{TC}(\{(\bar{x}, \bar{y}) \mid \mathcal{A} \vDash \varphi(\bar{x}, \bar{y}, \bar{b}\}),$$

where for any binary relation R, $\mathrm{TC}(R) = \{(a, b) \mid$ There exist $e_0, \ldots, e_n$ such that $e_0 = a$, $e_n = b$ and for all $i < n$, $R(e_i, e_{i+1})\}$.

For any binary relation R, $\mathrm{DTC}(R) = \{(a, b) \mid$ There exist $e_0, \ldots, e_n$ such that $e_0 = a$, $e_n = b$ and for all $i < n$, e_{i+1} is the unique e such that $R(e_i, e)\}$. The logic FO(DTC) is defined just like FO(TC), with TC replaced by DTC.

Example. A graph (G, E^G) is connected iff it is a model of the sentence $\forall x \forall y (x \neq y \rightarrow [\mathrm{TC}_{x,y} Exy]xy$.

3.-4.Vorlesungen

Turing machines and complexity classes

A *Turing machine* M is a finite device equipped with a tape which is bounded to the left and unbounded to the right. The tape is divided into cells. At each stage of computation each cell is either blank or contains a single symbol from a finite nonempty alphabet $\mathbb{A}$ that is fixed in advance. There is however one exception: the leftomost cell contains the symbol α, which does not belong to $\mathbb{A}$. M has a head which at each stage scans a single cell and in any computation step either erases or replaces the symbol it sees by another symbol and either moves one cell to the left or right or remains at the same cell.

M is also equipped with a finite set of *states*, $\mathrm{State}(M)$. There is an *initial state* s_0 as well as an *accepting state* s_+ and a *rejecting state* s_-, which are distinct. The action of M at a computation stage depends on its current state and on the symbol being scanned by its head. It is specified by a finite set of *instructions* $\mathrm{Instr}(M)$, consisting of expressions of the form

$$sa \to s'bh$$

where:

- s, s' are states, s is neither s_+ nor s_-
- a, b belong to $\mathbb{A} \cup \{\alpha, \mathrm{blank}\}$ and ($a = \alpha$ iff $b = \alpha$)
- $h \in \{-1, 0, 1\}$ and if $a = \alpha$ then $h \neq -1$

The expression $sa \to s'bh$ means: if M is in state s scanning the symbol a then it replaces a by b, its head moves one cell to the left (if $h = -1$), one cell to the right (if $h = 1$) or stays put (if $h = 0$) and then goes into state s'.

M is *deterministic* if for all states s and symbols $a \in \mathbb{A} \cup \{\alpha, \mathrm{blank}\}$ there is at most one instruction of the above form in $\mathrm{Instr}(M)$. Machines that are not required to be deterministic are said to be *nondeterministic.*

$\mathbb{A}^*$ denotes the set of words over $\mathbb{A}$ and $\mathbb{A}^+$ the set of nonempty such words. If $u = a_1 \cdots a_r$ belongs to $\mathbb{A}^*$ then M is *started with* u if M begins a computation (or run) in the initial state s_0 with its head on the leftmost cell (scanning α) and with the symbols $a_1, \ldots, a_r$ written in order on the first r cells to the right of the leftmost cell, followed by blanks. M then computes by implementing one of its instructions at each step, until there is no instruction telling it what to do next. If M is then in the state s_+ then we have an *accepting run* and if in the state s_- a *rejecting run.* M *accepts* u if there is at least one accepting run with input u and M *rejects* u if all runs with input u are finite and rejecting.

A *language* is a subset of $\mathbb{A}^+$. A language L is *accepted* by M if for all $u \in \mathbb{A}^+$, M accepts u iff $u \in L$. L is *decided* by M iff in addition M rejects u iff $u \notin L$. L is *decidable* if it is decided by some deterministic Turing machine and is *(effectively) enumerable* if it is accepted by some nondeterministic Turing machine.

Suppose $f : \mathbb{N} \to \mathbb{N}$. Then M is f *time-bounded* if for all u accepted by M there is an accepting run of M with input u which has length at most $f(|u|)$, where $|u|$ is the length of the word u. M is f *space-bounded* if for all u accepted by M there is an accepting run of M with input u which uses at most $f(|u|)$ cells before stopping. L is in PTIME (PSPACE respectively) if it is accepted by a deterministic M that is p time-bounded (p space-bounded, respectively) where p is given by a polynomial. The classes NPTIME (NPSPACE) are defined in the same way, but allowing nondeterministic machines. One can show the following:

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

We will also consider the complexity classes LOGSPACE and NLOGSPACE, in which the work space needed by the machine is less than the length of the input; this will be clarified later when we introduce Turing machines with both an input tape and work tapes.

Ordered Structures as Inputs

We want to compute with finite structures as inputs and for this purpose we need to code such structures as finite strings in some alphabet. This is easier for ordered structures, which are defined as follows.

Fix the vocabulary $\tau_0 = \{<, S, \min, \max\}$ where $<, S$ are binary relation symbols and min, max are constant symbols. Let τ be a vocabulary containing τ_0. A τ-structure $\mathcal{A}$ is *ordered* if the reduct $\mathcal{A} \upharpoonright \tau_0$ is an ordering (i.e., it interprets $<$ as a total ordering, S as the successor relation of this ordering and $\min, \max$ as the least and greatest elements of this ordering). The class of ordered τ-structures is denoted by $\mathcal{O}[\tau]$. If ψ is a sentence in the language of τ then $\mathrm{ordMod}(\psi)$ denotes the class of ordered models of ψ.

Write the vocabulary τ as $\tau_0 \cup \{R_1, \ldots, R_k, c_1, \ldots, c_l\}$ where the R_i's are relation symbols and the c_j's are constant symbols. We define a *Turing machine M for τ-structures* as follows. M has $1+k+l$ *input tapes* and m *work tapes* for some positive m. All tapes are bounded to the left, unbounded to the right and their cells are numbered by $-1, 0, 1, \ldots$, starting with the leftmost cell which contains the symbol α. Each input tape contains an input word followed by the symbol ω, indicating the end of the input. Each tape has its own head, and these heads move independently of each other. The input heads are read-only, while the work heads are read-and-write. The alphabet contains only the single symbol "1" and we identify "blank" with "0".

Now suppose that $\mathcal{A}$ is an ordered τ-structure with universe $\{0, 1, \ldots, n-1\}$. We write $\mathcal{A}$ as an input to the machine M as follows. Order the $1+k+l$ input tapes from 0 to $k+l$. The 0-th tape contains a sequence of 1's of length n and therefore as α in cell -1, 1 in cells 0 through $n-1$ and ω in cell n, followed by 0's to the right. For $1 \leq i \leq k$ the i-th input tape contains information about the relation $R = R_i^{\mathcal{A}}$, coded as follows: Let r be the arity of R; for $j < n^r$ the j-th cell will contain "1" just in case the j-th r-tuple in the lexicographic ordering of $\{0, 1, \ldots, n-1\}^r$ belongs to R. For $1 \leq i \leq l$, the $(k+i)$-th input tape contains the binary representation of $c_i^{\mathcal{A}}$ without leading 0's.

Computations run as follows. M starts in its initial state with $\mathcal{A}$ written on the input tapes as above, with the work tapes empty and with all heads positioned at cell 0. Instructions now take the form

$$s b_0 \cdots b_{k+l} c_1 \cdots c_m \to s' c_1' \cdots c_m' h_0 \cdots h_{k+l+m}$$

with the meaning: If M is in state s with its heads scanning $b_0, \ldots, b_{k+l}$ on the input tapes and $c_1, \ldots, c_m$ on the work tapes, then M replaces the c_i by the c_i', moves the i-th head according to h_i and enters state s'. Of course the b_i's belong to $\{0, 1, \alpha, \omega\}$, the c_i's and c_i''s belong to $\{0, 1, \alpha\}$ and the h_i's belong to $\{-1, 0, 1\}$, with the obvious restrictions. M is *deterministic* if any two instructions with the same left half (part before the $\to$) have the same right half (part after the $\to$). If we don't require determinism, we use the term *nondeterministic.*

For $f : \mathbb{N} \to \mathbb{N}$ we say that M is f *time-bounded* if for any $\mathcal{A}$ accepted by M there is a run accepting $\mathcal{A}$ of length at most $f(n)$, where $\{0, 1, \ldots, n-1\}$ is the domain of $\mathcal{A}$. And M is f *space-bounded* if for all $\mathcal{A}$ accepted by M there is a run which accepts $\mathcal{A}$ and uses at most $f(n)$ cells on each work tape before stopping.

If K is a class of ordered τ-structures then M *accepts* K if M accepts exactly those ordered τ-structures in K. We define PTIME, NPTIME, PSPACE and NPSPACE in the natural way. Also, we define: K is in LOGSPACE (NLOGSPACE, respectively) if there is a deterministic (nondeterministic, respectively) M and $d \geq 1$ such that M accepts K and is $d \cdot \log$ space-bounded (where $\log n$ is the least natural number $\geq \log_2 n$).

5.-6.Vorlesungen

Logical Descriptions of Computations

Recall that our goal is to show that for a class K of ordered structures we have:

$K \in$ 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

In this section we prove the implications from left to right. Our approach is the following: For any of the above complexity classes $\mathcal{C}$, let M be a Turing machine (for τ-structures, where τ is a finite relational vocabulary containing τ_0) which witnesses that the class K belongs to $\mathcal{C}$. We will construct a formula φ_M in the logic $\mathcal{L}$ associated as above with the complexity class $\mathcal{C}$ so that for any ordered structure $\mathcal{A}$,

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

It follows that $K = \mathrm{ordMod}(\varphi_M)$, as desired.

The formula φ_M is built by translating the configurations and transitions between them in an M-computation into first-order logic; the additional non-first-order power of the relevant logics is used to determine when the computation stops and to derive its outcome.

For simplicity assume that our vocabulary τ is $\tau_0 \cup \tau_1$ where τ_0 is $\{<, S, \min, \max\}$, τ_1 is disjoint from τ_0 and τ_1 consists just of relation symbols $\{R_1, \ldots, R_k\}$ where R_i is r_i-ary. For convenience set $r_0 = 1$.

Suppose that M is a Turing machine for τ-structures, with $1 + k$ input tapes and m work tapes. A *configuration of M* consists of the following information:

The current state;
What is currently written on the work tapes;
The current positions of all input and work tape heads

A configuration is *accepting* if its state is s_+. A configuration CONF$'$ is *a successor to* the configuration CONF if an instruction of M allows M to pass from configuration CONF to configuration CONF$'$ in one step. For convenience, we allow any accepting configuration to be a successor to itself.

We first consider the case of PSPACE. Suppose that M is a Turing machine for τ-structures which is x^d space-bounded, i.e., if M accepts an ordered structure $\mathcal{A}$ then there is an accepting run with input $\mathcal{A}$ that scans at most n^d cells on each work tape, where n is the size of the universe of $\mathcal{A}$. We make the following simplifying assumptions: We assume that d is at least r_i, the arity of the relation symbol R_i, for each i and that n, the size of the universe of $\mathcal{A}$, is greater than both $k + m$ and the number of states. Also assume that State(M) is an initial segment of $\mathcal{N}$ with $s_0 = 0$ as the starting state.

To code the data of CONF we introduce the "state relation" $\mathrm{ST}^{\mathrm{CONF}}$, the "end-of-tape relations" E_j^{CONF}, the "head relations" H_j^{CONF} and the "inscription relations" I_j^{CONF}. Here are the definitions:

$\mathrm{ST}^{\mathrm{CONF}} = \{s\}$ where s is the state of CONF.

For $0 \leq j \leq k + m$:

$E_j^{\mathrm{CONF}} = \{0\}$ if the j-th head is scanning α
$E_j^{\mathrm{CONF}} = \{n-1\}$ if the j-th head is scanning ω
$E_j^{\mathrm{CONF}} = \emptyset$ otherwise.

For $0 \leq j \leq k$:

$H_j^{\mathrm{CONF}} = \{|e|_{r_j} \mid 0 \leq e$, the j-th head scans the e-th cell and this cell does not contain $\omega\}$. ($|e|_{r_j}$ denotes the e-th r_j-tuple from $\{0, 1, \ldots, n-1\}$.)

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

$H_j^{\mathrm{CONF}} = \{|e|_d \mid 0 \leq e$, the j-th head scans the e-th cell$\}$. (Note that only the first n^d cells can be scanned.)

$I_j^{\mathrm{CONF}} = \{|e|_d \mid 0 \leq e < n^d$ and the e-th cell of the j-th tape contains the symbol $1\}$.

Note that as we have assumed that $r_i \leq d$ for all i, all of the above relations are d_0-ary for some $d_0 \leq d$. We put all of these relations into a single $(d+2)$-ary relation C^{CONF} as follows:

$C^{\mathrm{CONF}} =$
$\{(0,0)\} \times \{\tilde{0}\} \times \mathrm{ST}^{\mathrm{CONF}} \cup$
$\bigcup_{0 \leq j \leq k+m} \{(1,j)\} \times \{\tilde{0}\} \times E_j^{\mathrm{CONF}} \cup$
$\bigcup_{0 \leq j \leq k+m} \{(2,j)\} \times \{\tilde{0}\} \times H_j^{\mathrm{CONF}} \cup$
$\bigcup_{k+1 \leq j \leq k+m} \{(3,j)\} \times I_j^{\mathrm{CONF}}$.

In the above, $\tilde{0}$ denotes a string of 0's of sufficient length to ensure arity $d+2$.

The following technical lemma will enable us to finish the cases of PSPACE, PTIME and NPTIME.

Lemma 2 *There are a first-order formula $\varphi_{start}(\bar{x})$ and second-order formulas $\varphi_{succ}(\bar{x}, X)$, $\psi_{succ}(X, Y)$ (without second-order quantifiers) such that for all ordered τ-structures $\mathcal{A}$ of large enough size n and all $\bar{a} \in A^{d+2}$ we have:*

(a) If C_0 denotes the starting configuration of M with input $\mathcal{A}$ then

$$\mathcal{A} \models \varphi_{start}[\bar{a}] \text{ iff } \bar{a} \in C_0.$$

(b) If M is deterministic and C is an n^d-bounded configuration of M then

$$\mathcal{A} \vDash \varphi_{succ}[\bar{a}, C] \text{ iff } C \text{ has an } n^d\text{-bounded successor } C' \text{ and } \bar{a} \in C'.$$

(c) If C_1 is an n^d-bounded configuration of M and C_2 a $(d+2)$-ary relation on A then

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

Now we can prove:

Theorem 3 *If K is a class of ordered τ-structures in PSPACE then K is axiomatisable in FO(PFP).*

Proof. Let M be a deterministic Turing machine witnessing $K \in$ PSPACE. Choose a suitable d so that M is x^d space-bounded. Define:

$\varphi(\bar{x}, X) \;\equiv$
$(\sim \exists \bar{y} X \bar{y} \wedge \varphi_{start}(\bar{x})) \vee (\exists \bar{y} X \bar{y} \wedge \varphi_{succ}(\bar{x}, X)),$

where φ_{start} and φ_{succ} are as in the preceding lemma. Let $\mathcal{A}$ be an ordered structure of size n. Then $F_0^\varphi, F_1^\varphi, \ldots$ is the sequence $\emptyset, C_0, C_1, \ldots$ 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 $C_i = C_{i+1}$;
If C_i is an n^d-bounded configuration without an n^d bounded successor configuration then $C_{i+1} = \emptyset$, $C_{i+2} = C_0$, $C_{i+3} = C_1, \cdots$ and therefore this sequence has no fixed-point.

In conclusion:

M accepts $\mathcal{A}$ iff
F_∞^φ is an accepting configuration iff
F_∞^φ is a configuration with state s_+.

The latter is expressed by the formula

$\exists y((y \text{ is the } s_+\text{-th element of } <) \wedge [\mathrm{PFP}_{\bar{x},X}\varphi] \widetilde{\min \min \min}\, y).$

So $K = \mathrm{ordMod}(\psi)$ where ψ is the above formula. □

Theorem 4 *If K is a class of ordered τ-structures in PTIME then K is axiomatisable in FO(IFP).*

Proof. Let M be an x^d time-bounded Turing machine witnessing that K is in PTIME (for suitable d). For $\bar{v} = v_0 \dots v_{d-1}$ we set

$$\varphi(\bar{v}, \bar{x}, Z) \equiv (\bar{v} = \widetilde{\min} \wedge \varphi_{start}(\bar{x})) \vee \exists \bar{u}(S^d \bar{u}\bar{v} \wedge \varphi_{succ}(\bar{x}, Z\bar{u}-)).$$

In this formula, $\bar{v} = \widetilde{\min}$ abbreviates $v_0 = \min \wedge \cdots \wedge v_{d-1} = \min$; $S^d\bar{u}\bar{v}$ stands for "$\bar{v}$ is the successor of $\bar{u}$ in the lexicographic ordering of d-tuples"; and $\varphi_{succ}(\bar{x}, Z\bar{u}-)$ is obtained from $\varphi_{succ}(\bar{x}, X)$ by replacing subformulas of the form $X\bar{t}$ by $Z\bar{u}\bar{t}$. Then for structures $\mathcal{A}$ of size n we have:

$\mathcal{A} \in K$ iff M accepts $\mathcal{A}$ iff
The $(n^d - 1)$-st configuration of M with input $\mathcal{A}$ is defined and has state s_+
iff
$\mathcal{A} \models [\mathrm{IFP}_{\bar{v}\bar{x},Z}\varphi]\widetilde{\max}\min\min\widetilde{\min}s_+$.

So K is the class of ordered models of a sentence in FO(IFP). □

Theorem 5 *If K is in NPTIME then K is axiomatisable by a Σ^1_1 sentence.*

Proof. Choose M witnessing that K is in NPTIME and d so that M is x^d time-bounded. Then for input structures $\mathcal{A}$ of size n:

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

where φ is the sentence:

$$\varphi \equiv \exists Z(\forall \bar{x}(Z\widetilde{\min}\bar{x} \leftrightarrow \varphi_{start}(\bar{x})) \wedge \forall \bar{u}\forall \bar{v}(S^d\bar{u}\bar{v} \rightarrow \psi_{succ}(Z\bar{u}-, Z\bar{v}-)) \wedge Z\widetilde{\max}\min\min\widetilde{\min}s_+).$$

This is a Σ^1_1 sentence. □

7.Vorlesung

LOGSPACE and NLOGSPACE

To capture these complexity classes we need to use the weaker logics FO(DTC) and FO(TC). Recall that these are defined as follows. To first order logic we add the generating rules:

If φ is a formula, $\bar{x}$ and $\bar{y}$ are sequences of variables (with no variable occurring twice), the tuples $\bar{s}$ and $\bar{t}$ are sequences of variables or constants and $\bar{x}, \bar{y}, \bar{s}, \bar{t}$ all have the same length then $[\mathrm{TC}_{\bar{x},\bar{y}}\varphi]\bar{s}\bar{t}$ is a formula of FO(TC) ($[\mathrm{DTC}_{\bar{x},\bar{y}}\varphi]\bar{s}\bar{t}$ is a formula of FO(DTC), respectively).

The meanings of these new formulas are given by.

$[\mathrm{TC}_{\bar{x},\bar{y}}\varphi(\bar{x},\bar{y},\bar{u})]\bar{s}\bar{t}$ iff $(\bar{s},\bar{t})$ belongs to the transitive closure TC of $\{(\bar{x},\bar{y}) \mid \varphi(\bar{x},\bar{y},\bar{u})\}$.

$[\mathrm{DTC}_{\bar{x},\bar{y}}\varphi(\bar{x},\bar{y},\bar{u})]\bar{s}\bar{t}$ iff $(\bar{s},\bar{t})$ belongs to the deterministic transitive closure DTC of $\{(\bar{x},\bar{y}) \mid \varphi(\bar{x},\bar{y},\bar{u})\}$.

Recall that for a binary relation R, $\mathrm{TC}(R)$ consists of pairs (a,b) such that for some $e_0,\dots,e_n$, $e_0 = a$, $e_n = b$ and for each $i < n$, (e_i, e_{i+1}) belongs to R. $\mathrm{DTC}(R)$ is defined in the same way except we require that e_{i+1} is the unique e such that (e_i, e) belongs to R.

A class K of ordered τ-structures belongs to LOGSPACE (NLOGSPACE) iff there is a deterministic (non-deterministic) Turing machine M and d such that M accepts K and is *$d \cdot log$ space-bounded*, i.e., each work tape scans at most $d \cdot log\ n$ cells during the course of the accepting run where n is the size of the input. As numbers less than n have binary-length at most $log\ n$, it follows that we can use d variables to code the contents of a work tape. And each head position can be represented by a number less than n (assuming n is larger than $d \cdot log\ n$). Therefore we can describe the data of a configuration by a sequence of length independent of n consisting of naturalnumbers less than n. The basic lemma providing formulas which describe the configurations of the machine is as follows.

Lemma 6 *Let M be $d \cdot log$ space-bounded. Then there are formulas $\chi_{start}(\bar{x})$ in FO and $\chi_{succ}(\bar{x}, \bar{x}')$ of FO(DTC) such that for ordered τ-structures $\mathcal{A}$ of large enough size n and $\bar{a}$ in A:*

(a) $\mathcal{A} \vDash \chi_{start}[\bar{a}]$ iff $\bar{a}$ is (a code for) the starting configuration.
(b) For any $d \cdot log$ n-bounded configuration $\bar{a}$ and any $\bar{b}$:

$\mathcal{A} \vDash \chi_{succ}[\bar{a}, \bar{b}]$ iff $\bar{b}$ is a $d \cdot log$ n-bounded successor configuration of $\bar{a}$.

The need for a formula of FO(DTC) in part (b) of the lemma is due to the fact that to express the $d \cdot log$ n-boundedness of a configuration we need to define addition, multiplication and binary exponentiation ($2^a = b$); the latter is done using deterministic transitive closure. For example:

$x + y = z$ iff $(y = \min \wedge z = x) \vee [\mathrm{DTC}_{uv,u'v'}(Suu' \wedge Svv')] \min xyz.$

Now we can show that LOGSPACE and NLOGSPACE are captured by the logics FO(DTC) and FO(TC), respectively.

Theorem 7 *If K is in LOGSPACE then K is axiomatisable in FO(DTC).*

Proof. Let M be a detrministic machine witnessing $K \in$ LOGSPACE which is $d \cdot log$ space-bounded. Let χ_{start} and χ_{succ} be the formulas corresponding to M according to the preceding lemma. Then we have, for a structure $\mathcal{A}$ of size n:

M accepts $\mathcal{A}$ iff
There is a sequence $\bar{a}_0, \ldots, \bar{a}_k$ of $d \cdot log$ n-bounded configurations such that $\bar{a}_0$ is the starting configuration, $\bar{a}_{i+1}$ is the successor configuration of $\bar{a}_i$ and $\bar{a}_k$ is an accepting configuration iff
$\mathcal{A} \vDash \exists \bar{x}(\chi_{start}(\bar{x}) \wedge \exists \bar{x}'([\mathrm{DTC}_{\bar{x},\bar{x}'}(\bar{x}, \bar{x}')]\bar{x}\bar{x}' \wedge x_1' = s_+)),$

where x_1' denotes the state of the configuration $\bar{x}'$. □

Theorem 8 *If K is in NLOGSPACE then K is axiomatisable in FO(TC).*

Proof. Just replace DTC by TC in the previous proof. □

8.-9.Vorlesungen

The complexity of the satisfaction relation

We have shown that if a class K of finite ordered structures belongs to the complexity class LOGSPACE then it is axiomatised by a formula of the logic DTC, first-order logic enhanced with a deterministic transitive closure quantifier. Now we prove the converse of this result, as well as similar results for the classes NLOGSPACE, PTIME, NPTIME and PSPACE.

Thus if φ is a sentence of DTC we want to show that the relation $\mathcal{A} \vDash \varphi$ can be decided by a machine M which is $d \cdot log$ space-bounded for some d; i.e., φ has a log-space *model checker*. In fact, we shall obtain a machine M which *strongly witnesses* that $K = \mathrm{ordMod}(\varphi)$ belongs to LOGSPACE, i.e.

M accepts K;
For any $\mathcal{A}$, every run of M with input $\mathcal{A}$ stops in state s_+ or s_-;
For any $\mathcal{A}$, every run of M with input $\mathcal{A}$ is log-space bounded.

Strong witnessing is defined analogously for the other complexity classes, with "log-space" replaced by "polynomial-time" or "polynomial-space", accordingly.

The proof is by induction on φ and therefore we need to conider not just sentences φ but also formulas $\varphi(x_1, \ldots, x_l, Y_1, \ldots, Y_r)$. For such a formula we define:

$\mathrm{ordMod}(\varphi) = \{(\mathcal{A}, a_1, \ldots, a_l, P_1, \ldots, P_r) \mid \mathcal{A} \vDash \varphi[\bar{a}, \bar{P}]\}.$

Theorem 9 *If a class of ordered structures is axiomatised in FO(DTC) then it is in LOGSPACE.*

Proof. By induction on the formula φ doing the axiomatisation.

φ atomic. Assume for simplicity that φ is Rxy. We want a machine M strongly witnessing that $\{(\mathcal{A}, i, j) \mid R^A ij\}$ belongs to LOGSPACE. View $(\mathcal{A}, i, j)$ as a τ'-structure, where $\tau' = \tau \cup \{c, d\}$ and c, d are new constant symbols. Assume that A is $\{0, 1, \ldots, n-1\}$. Then whether or not $R^A ij$ holds is coded in the $(i \cdot n + j)$-th square of the input tape corresponding to the relation symbol R and the binary representations of i, j are available on the input tapes corresponding to the constant symbols c, d. Using this we can design the machine M strongly witnessing that $\mathrm{ordMod}(Rxy)$ belongs to LOGSPACE.

$\varphi = \sim \psi$. By induction there is a machine M strongly witnessing that ordMod(ψ) belongs to LOGSPACE. To get such a machine for φ, just switch the states s_+ and s_-.

$\varphi = (\psi \vee \chi)$. Apply induction to get machines M_ψ and M_χ for ψ, χ, respectively. Let M be the machine that first carries out the computation for M_ψ, erases the work tapes, and then carries out the computation for M_χ, accepting the input if at least one of M_ψ, M_χ accepts.

$\varphi(x_1, \dots, x_l) = \exists x \psi$. By induction we can choose a machine M_0 for $\psi(x_1, \dots, x_l, x)$. The desired machine M for φ proceeds as follows: Suppose that the input is $(\mathcal{A}, a_1, \dots, a_l)$ where $A = \{0, 1, \dots, n-1\}$. Then for $i = 0, 1, \dots, n-1$, M writes the binary representation of i on a work tape and checks, using M_0, whether $\mathcal{A} \vDash \psi[a_1, \dots, a_l, i]$. If the answer is positive at least once, M stops in state s_+; otherwise M stops in state s_-.

$\varphi = [DTC_{\bar{x},\bar{y}}\psi]\bar{s}\bar{t}$. For simplicity, assume that the free variables of ψ are among $\bar{x}, \bar{y}$ and that $\bar{x} = x$, $\bar{y} = y$, $\bar{s} = s$, $\bar{t} = t$. Choose a machine M_0 strongly witnessing that ordMod(ψ) belongs to LOGSPACE. Given $\mathcal{A}$ with $A = \{0, 1, \dots, n-1\}$, if there is a ψ-path from s to t then there is one of length at most n. Therefore, the desired machine M can be designed as follows: It writes s on a work tape andsets a counter to n, 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 $j = 0, 1, \dots, n-1$ whether $\mathcal{A} \vDash \psi[s, j]$ holds for exactly one j; if not, M rejects. If so, M checks whether j equals t; if so, M accepts, and otherwise M replaces s by j and reduces the counter by one. □

The previous proof also gives a partial result about NLOGSPACE. Define FO(posTC) to be the formulas of FO(TC) which only contain positive occurrences of TC, i.e., each such occurrence is in the scope of an even number of negation symbols. By introducing $\wedge$ and $\forall$, we can in fact assume that in such formulas, TC does not occur in the scope of any negation symbol.

Theorem 10 *If a class of ordered structures is axiomatised in FO(posTC) then it is in NLOGSPACE.*

Proof. The proof is just as in the previous theorem, noting that in the case $\varphi = \sim \psi$, we can assume that ψ does not contain TC and therefore is first-order. □

We will show later that FO(posTC) has the full expressive power of FO(TC) and therefore can be replaced by FO(TC) in the statement of the previous theorem.

Theorem 11 *If a class of ordered structures is axiomatised in FO(IFP) then it is in PTIME.*

Proof. Again by induction on φ. The atomic case as well as the negation, disjunction and existential quantification cases are just as in the previous proof.

Suppose that φ is $[\mathrm{IFP}_{\bar{x},X}\psi(\bar{x},X)]\bar{t}$ where X is r-ary. Let M_0 be a machine strongly witnessing that $\{(\mathcal{A},\bar{a},R) \mid \mathcal{A} \vDash \psi[\bar{a},R]\}$ belongs to PTIME. The desired machine M contains a subroutine which uses work tapes W and W'. If it begins with a word of length n^r on W which codes an r-ary relation R and nothing on W', then the subroutine uses M_0 to write a code for $R' = \{\bar{a} \mid \mathcal{A} \vDash (X\bar{x} \vee \psi)[\bar{a},R]\}$ on the work tape W', leaving the contents of W unchanged. Now M runs as follows: It sets $R = \emptyset$ and uses the subroutine to calculate R'. If $R = R'$ it checks if $R\bar{t}$ and accepts or rejects accordingly. Otherwise, it sets R to be R', erases the contents of W' and again applies the subroutine. After at most n^d calls to the subroutine, $R = R'$ will be achieved. □

A similar argument gives:

Theorem 12 *If a class of ordered structures is axiomatised in FO(PFP) then it is in PSPACE.*

Proof. As in the previous proof we only treat the fixed-point quantifier case. Suppose that φ is $[\mathrm{PFP}_{\bar{x},X}\psi(\bar{x},X)]\bar{t}$ where X is r-ary and M_0 is a machine strongly witnessing that $\{(\mathcal{A},\bar{a},R) \mid \mathcal{A} \vDash \psi[\bar{a},R]\}$ belongs to PSPACE. The operator F^ψ satisfies $F^\psi_{2^{n^r}-1} = F^\psi_{2^{n^r}}$ (and this is F^ψ_∞) or F^ψ_∞ is empty. The desired machine M starts its computation on the input $\mathcal{A}$ by setting a counter to $2^{n^r}-1$, by writing the word $11\cdots 1$ of length n^r on a work tape and then it proceeds as in the IFP case, but now using the counter to ensure that the subroutine which now evaluates $R' = \{\bar{a} \mid \mathcal{A} \vDash \psi[\bar{a},R]\}$ is invoked at most $2^{n^r}-1$ times. When the counter goes negative, M checks whether $R = R'$ and whether $R\bar{t}$ holds; if both answers are positive it accepts, otherwise it rejects. □

Also note:

Theorem 13 *If a class of ordered structures is axiomatised in SO (second-order logic) then it is in PSPACE.*

Proof. By induction on the formula φ doing the axiomatising; as before we need only handle the second-order quantifier case $\varphi = \exists X \psi$. Suppose that X is r-ary and let M_0 be a machine strongly witnessing that $\{(\mathcal{A}, R) \mid \mathcal{A} \vDash \psi[R]\}$ belongs to PSPACE. Given a structure $\mathcal{A}$ of size n, the desired machine M writes the word $11 \cdots 1$ of length n^r on a work tape W and repeatedly decreases this word, using the machine M_0 to check in each case if ψ holds when X is interpreted by the contents of the tape W. $\square$

Theorem 14 *If a class of ordered structures is axiomatised in Σ_1^1 then it is in NPTIME.*

Proof. Suppose that the given class is axiomatised by $\varphi = \exists X_1 \cdots \exists X_l \psi$ where ψ is first-order and the arity of X_i is r_i. Let M_0 be a machine strongly witnessing that $\mathrm{ordMod}(\psi(X_1, \ldots, X_l))$ is in PTIME. The desired machine M proceeds as follows on input $\mathcal{A}$: it nondeterministically writes words over $\{[0, 1\}$ of length n^{r_1}, ..., n^{r_l} on different work tapes, coding interpretations $P_1, \ldots, P_l$ of $X_1, \ldots, X_l$. Then, using M_0, it checks whether $\mathcal{A} \vDash \psi[P_1, \ldots, P_l]$ or not, and stops in an accepting or rejecting state correspondingly. $\square$

To summarise, we have the following so far:

Theorem 15 *Let K be a class of ordered structures. Then:*
$K \in$ LOGSPACE iff K is axiomatised in FO(DTC).
If $K \in$ NLOGSPACE then K is axiomatised in FO(TC), and if K is axiomatised in FO(posTC) then $K \in$ NLOGSPACE.
$K \in$ PTIME iff K is axiomatised in FO(IFP).
$K \in$ NPTIME iff K is axiomatised in Σ_1^1.
$K \in$ PSPACE iff K is axiomatised in FO(PFP) (and this includes all K axiomatised in SO).

10.-11.Vorlesungen

To obtain a descriptive complexity-theoretic characterisation of the complexity class NLOGSPACE we need the following.

Theorem 16 *If K is axiomatised in FO(TC) then it is also axiomatised in FO(posTC).*

Recall that the logic FO(TC) is obtained by adding formulas of the form $[\mathrm{TC}_{\bar{x},\bar{x}'}\psi(\bar{x},\bar{x}')]\bar{s}\bar{t}$, where $\bar{x},\bar{x}'$ are sequences of variables, $\bar{s},\bar{t}$ are sequences of terms, all sequences have the same length and ψ is a formula already belonging to the logic. The intended meaning is that there is a sequence $\bar{s}=\bar{s}_0, \bar{s}_1, \ldots, \bar{s}_n=\bar{t}$ such that $\psi(\bar{s}_i,\bar{s}_{i+1})$ for each $i<n$.

First-order logic can be formulated using $\sim$, $\vee$ and $\exists$; for our present purposes it will be useful to add $\wedge$ and $\forall$, and restrict $\sim$ to only occur in front of an atomic formula.

Thus any formula of FO(posTC), in which TC occurs only positively, is built from atomic and negatomic (negations of atomic) formulas, using $\forall$, $\exists$, $\vee$, $\wedge$ and TC. Our first goal is to show that on ordered structures we can in fact eliminate both $\forall$ and $\exists$ and allow at most one use of TC (at the front of the formula).

Lemma 17 *On structures in a vocabulary τ containing $\{<,S,\min,\max\}$, every FO(posTC) formula is equivalent (on models with at least two elements) to a formula of the form $[TC_{\bar{x},\bar{x}'}\psi(\bar{x},\bar{x}')]\widetilde{\min}\widetilde{\max}$ where ψ is quantifier-free.*

Proof. First note that $\forall x\varphi(x)$ is equivalent to

$$[\mathrm{TC}_{x,y}(\varphi(x)\wedge Sxy\wedge\varphi(y))]\min\max\,.$$

This eliminates the universal quantifier. Also note that on finite structures, $\exists z\varphi(z)$ is equivalent to

$$[\mathrm{TC}_{x,y}((x=\min\wedge\varphi(y))\vee(\varphi(x)\wedge y=\max))]\min\max\,.$$

This eliminates the existential quantifier.

Now we show that any formula φ built from atomic and negatomic formulas using just $\vee$, $\wedge$ and TC is equivalent to a formula of the desired form, by induction on φ. For simplicity of notation, write min as c and max as d. Thus $\widetilde{\min},\widetilde{\max}$ will be written simply as $\bar{c},\bar{d}$.

The cases of atomic and negatomic φ are easy, as φ is equivalent to $[\mathrm{TC}_{\bar{x},\bar{x}'}\varphi]\bar{c}\bar{d}$ where $\bar{x},\bar{x}'$ are variables not occurring in φ.

Suppose that φ is $\varphi_1 \vee \varphi_2$ where φ_i is equivalent to $[\mathrm{TC}_{\bar{x},\bar{x}'}\psi_i(\bar{x},\bar{x}')]\bar{c}\bar{d}$, ψ_i quantifier-free. (We can assume that $\bar{x},\bar{x}'$ are the same for φ_1 and φ_2.) Then φ is equivalent to:

$$[\mathrm{TC}_{\bar{x}x,\bar{x}'x'}\psi(\bar{x}x,\bar{x}'x')]\bar{c}c\bar{d}d$$

where:

$\psi(\bar{x}x,\bar{x}'x') \equiv$
$(x = x' = c \wedge \psi_1)$
$\vee(\bar{x} = \bar{x}' = \bar{d} \wedge x = c \wedge x' = d)$
$\vee(\bar{x} = \bar{x}' = \bar{c} \wedge x = c \wedge x' = d)$
$\vee(x = x' = d \wedge \psi_2)$.

Similarly, if $\varphi \equiv (\varphi_1 \wedge \varphi_2)$ where φ_1, φ_2 are of the above form then we take $\psi(\bar{x}x,\bar{x}'x')$ to express the existence of the following path:

$$\bar{c}c \ldots \rightarrow_{\psi_1} \ldots \bar{d}c \rightarrow \bar{c}d \ldots \rightarrow_{\psi_2} \ldots \bar{d}d.$$

Next suppose that φ is $[\mathrm{TC}_{\bar{u},\bar{v}}\varphi']\bar{s}\bar{t}$ and by induction write φ' as $[\mathrm{TC}_{\bar{x},\bar{x}'}\psi'(\bar{x},\bar{x}',\bar{u},\bar{v})]\bar{c}\bar{d}$ with ψ' quantifier-free. Let $\bar{s} = \bar{e}_0, \ldots, \bar{e}_k = \bar{t}$ be a path witnessing that $(\bar{s},\bar{t})$ belongs to $\mathrm{TC}(\varphi'(\cdot,\cdot))$. Then for $i < k$ there is a ψ'-path witnessing that $(\bar{c},\bar{d})$ belongs to $\mathrm{TC}(\psi'(\cdot,\cdot,\bar{e}_i,\bar{e}_{i+1}))$. Therefore φ is equivalent to:

$$[\mathrm{TC}_{\bar{u}\bar{v}\bar{x},\bar{u}'\bar{v}'\bar{x}'}\psi(\bar{u},\bar{v},\bar{x},\bar{u}',\bar{v}',\bar{x}')]\bar{c}\bar{c}\bar{c}\bar{d}\bar{d}\bar{d},$$

where:

$\psi(\bar{u},\bar{v},\bar{u}',\bar{v}',\bar{x}') \equiv$
$(\bar{u} = \bar{c} \wedge \bar{v} = \bar{c} \wedge \bar{x} = \bar{c} \wedge \bar{u}' = \bar{s} \wedge \bar{x}' = \bar{c})$
$\vee(\bar{x} \neq \bar{d} \wedge \bar{u}' = \bar{u} \wedge \bar{v}' = \bar{v} \wedge \psi'(\bar{x},\bar{x}',\bar{u},\bar{v}))$
$\vee(\bar{x} = \bar{d} \wedge \bar{v} \neq \bar{t} \wedge \bar{u}' = \bar{v} \wedge \bar{x}' = \bar{c})$
$\vee(\bar{x} = \bar{d} \wedge \bar{v} = \bar{t} \wedge \bar{u}' = \bar{d} \wedge \bar{v}' = \bar{d} \wedge \bar{x}' = \bar{d})$.

In the formula ψ, the first line sets the corresponding starting values, the second line takes care of witnessing that $(\bar{c},\bar{d})$ belongs to $\mathrm{TC}(\psi'(\cdot,\cdot,\bar{c}_i,\bar{c}_{i+1})$, the next line allows one to pass from the tuples $\bar{e}_i,\bar{e}_{i+1}$ to $\bar{e}_{i+1},\bar{e}_{i+2}$ and the last line says that $\bar{e}_k$ equals $\bar{t}$. $\square$

Proof of Theorem 16. We show by induction that every formula of FO(TC) is equivalent to a formula of FO(posTC). The only nontrivial case is the negation case, and by Lemma 17 we may assume that our formula is of the form

$$\sim [\mathrm{TC}_{\bar{x},\bar{x}}\psi]\widetilde{\min}\widetilde{\max}$$

with ψ first-order (indeed quantifier-free).

For a structure $\mathcal{A}$ and $\bar{a}, \bar{b}$ from A^r let $d_\psi(\bar{a}, \bar{b})$ be the length of the shortest ψ-path connecting $\bar{a}$ and $\bar{b}$, setting this to be ∞ in case there is no such path. If $d_\psi(\bar{a}, \bar{b})$ is finite then its value is at most $|A|^r$. Our formula φ is equivalent to the assertion

$$|\{\bar{v} \mid d_{\psi(\bar{x},\bar{y})}(,\bar{v}) < \infty\}| = |\{\bar{v} \mid d_{\psi(\bar{x},\bar{y})\wedge\bar{y}\neq\widetilde{\max}}(\bar{s}, \bar{v}) < \infty\}|.$$

We shall use $|A|$-adic representations of natural numbers; in particular the number $|A|^r$ has the string $1\tilde{0}$ of length $r+1$ as its $|A|$-adic representation.

Claim 1. For $\varphi(x_1, \ldots, x_r, y_1, \ldots, y_r) \in \mathrm{FO(posTC)}$ there is an FO(posTC) formula $\chi(\bar{x}, \bar{y}, z_1, \ldots, z_{r+1})$ expressing the property $d_\varphi(\bar{x}, \bar{y}) \leq [\bar{z}]$, in the sense that for an ordered structure $\mathcal{A}$ and $\bar{a}, \bar{b}, \bar{m}$ in A

$$\mathcal{A} \vDash \chi[\bar{a}, \bar{b}, \bar{m}] \text{ iff } d_\varphi(\bar{a}, \bar{b}) \leq [\bar{m}],$$

where $[\bar{m}]$ denotes the number with $|A|$-adic representation $\bar{m}$.

To obtain χ, use the TC operator to go through all of the tuples lying on a path from $\bar{x}$ to $\bar{y}$:

$$\chi(\bar{x}, \bar{y}, \bar{z}) \equiv [\mathrm{TC}_{\bar{w}\bar{z},\bar{w}'\bar{z}'}(\varphi(\bar{w}, \bar{w}') \wedge [\bar{z}] < [\bar{z}'])]\bar{x}0\tilde{0}\bar{y}\bar{z}.$$

Claim 2. For first-order $\varphi(\bar{x}, \bar{y})$ there is an FO(posTC) formula $\rho_\varphi(\bar{x}, \bar{z})$ expressing the property $|\{\bar{y} \mid d_\varphi(\bar{x}, \bar{y}) < \infty\}| = [\bar{z}]$.

Given this, our formula $\sim [\mathrm{TC}_{\bar{x},\bar{y}}\psi(\bar{x}, \bar{y})]\widetilde{\min}\widetilde{\max}$ is equivalent to the FO(posTC) formula

$$\exists \bar{z}(\rho_{\psi(\bar{x},\bar{y})}(\widetilde{\min}, \bar{z}) \wedge \rho_{\psi(\bar{x},\bar{y})\wedge\bar{y}\neq\widetilde{\max}}(\widetilde{\min}, \bar{z})),$$

so we are done.

Proof of Claim 2. Consider the function

$$g(\bar{u}) = |\{\bar{y} \mid d_\varphi(\bar{x}, \bar{y}) \leq [\bar{u}]\}|.$$

Then $g(1\tilde{0})$ is $|\{\bar{y} \mid d_\varphi(\bar{x},\bar{y}) \leq |A|^r\}| = |\{\bar{y} \mid d_\varphi(\bar{x},\bar{y}) < \infty\}|$, the value we want to define. We give an inductive definition of g.

Suppose that $g(\bar{u}) = \bar{z}$ where $\bar{u} = u_1u_2\ldots u_{r+1}$. Fix $\bar{y}$. If $\bar{u} = 0\tilde{0}$ then $d_\varphi(\bar{x},\bar{y}) \not\leq [\bar{u}]+1$ is equivalent to $\sim \varphi(\bar{x},\bar{y})$. Otherwise $d_\varphi(\bar{x},\bar{y}) \not\leq [\bar{u}]+1$ iff there are $\bar{z}$-many $\bar{w}$ such that $\bar{w} \neq \bar{y}$, $d_\varphi(\bar{x},\bar{w}) \leq [\bar{u}]$ and $\sim \varphi(\bar{w},\bar{y})$. In the following formula $\chi_\varphi(\bar{x},\bar{u},\bar{z},\bar{z}')$ the second TC occurrence counts these $\bar{w}$ (with counting variables $\bar{q} = q_1\ldots q_{r+1}$) and therefore checks whether or not there are $\bar{z}$ many. If $g(\bar{u}) = \bar{z}$ then $\chi_\varphi(\bar{x},\bar{u},\bar{z},\bar{z}')$ expresses that $g(\bar{u}+1) = [\bar{z}']$ (where $\bar{u}+1$ denotes the string representing the number $[\bar{u}]+1$).

$\chi_\varphi(\bar{x},\bar{u},\bar{z},\bar{z}') \equiv$
$[\mathrm{TC}_{\bar{y}\bar{v},\bar{y}'\bar{v}'}(([\bar{y}'] = [\bar{y}]+1 \wedge [\bar{v}'] = [\bar{v}]+1 \wedge d_\varphi(\bar{x},\bar{y}) \leq [\bar{u}]+1)$
$\vee([\bar{y}'] = [\bar{y}]+1 \wedge \bar{v}' = \bar{v} \wedge \bar{u} = \tilde{0} \wedge \sim \varphi(\bar{x},\bar{y}))$
$\vee([\bar{y}'] = [\bar{y}]+1 \wedge \bar{v}' = \bar{v} \wedge \bar{u} \neq \tilde{0}$
$\wedge[\mathrm{TC}_{\bar{w}\bar{q},\bar{w}'\bar{q}'}(([\bar{w}'] = [\bar{w}]+1 \wedge \bar{q}' = \bar{q}) \vee ([\bar{w}'] = [\bar{w}]+1 \wedge [\bar{q}'] = [\bar{q}]+1$
$\wedge \bar{w} \neq \bar{y} \wedge d_\varphi(\bar{x},\bar{w}) \leq [\bar{u}] \wedge \sim \varphi(\bar{w},\bar{y})))]\tilde{0}0\tilde{0}\tilde{1}\bar{z}'$.

We can view χ_φ as an FO(posTC) formula. Now set

$$\rho_\varphi(\bar{x},\bar{z}) \equiv [\mathrm{TC}_{\bar{u}\bar{z},\bar{u}'\bar{z}'}([\bar{u}'] = [\bar{u}]+1 \wedge \chi_\varphi(\bar{x},\bar{u},\bar{z},\bar{z}'))]0\tilde{0}\tilde{0}01\tilde{0}\bar{z}.$$

This expresses that $g(1\tilde{0})$ equals $\bar{z}$ and hence that $|\{\bar{y} \mid d_\varphi(\bar{x},\bar{y}) < \infty\}| = [\bar{z}]$. And ρ_φ can be viewed as an FO(posTC) formula. □

12.Vorlesung

We have shown, for example, that on ordered structures, the logics FO(IFP) and FO(PFP) capture the complexity classes PTIME and PSPACE, respectively. However complexity classes are normally defined not with regard to collections of ordered structures, but with regard to collections of nonempty words on a finite alphabet. Fortunately we have:

Fact. There are translations from nonempty words into ordered structures and from ordered structures into nonempty words so that under these translations, a set or ordered structures or words belongs to a complexity class LOGSPACE, NLOGSPACE, PTIME, NPTIME, PSPACE iff the set of translations of its elements does.

Thus, for example, it follows that FO(IFP) has the same expressive power as FO(PFP) on ordered structures iff PTIME equals PSPACE in complexity theory. The same holds for the statement that PTIME equals NPTIME if we replace FO(PFP) by Σ^1_1. A fact in complexity theory is that LOGSPACE is stricly smaller than PSPACE; by the above this means that the logic FO(DTC) has less expressive power than the logic FO(PFP).

Now consider the capturing of complexity classes of unordered structures. If K is a class of unordered structures let $K_<$ denote the class of structures obtained by adding an ordering to the structures in K. We say that a logic $\mathcal{L}$ *strongly captures* the complexity class $\mathcal{C}$ iff for all classes K of unordered structures, $K_<$ belongs to $\mathcal{C}$ iff K is axiomatisable in $\mathcal{L}$. Then strong capturing implies capturing, but the converse is false: If K is the class of structures in the empty language of even size, then $K_<$ is axiomatised by a sentence of FO(DTC):

$$\sim [\mathrm{DTC}_{x,y} y = x + 2] \min \max .$$

But we shall show that no sentence of even FO(PFP) axiomatises K.

However if we consider Σ^1_1, whose expressive power is dominated by that of FO(PFP) only on *ordered* structures, we do get strong capturing, because if φ is a sentence with a relation symbol $<$ for an ordering, we can replace φ by the sentence $\psi \equiv \exists <\ \varphi$. Thus NPTIME is complemented iff Σ^1_1 has the same expressive power as Π^1_1 on arbitrary structures. The latter easily implies that if NPTIME is complemented then Σ^1_1 has the full expressive power of second-order logic and the polynomial-time hierarchy collapses to Σ^1_1.

Whether the classes LOGSPACE, NLOGSPACE or PTIME can be strongly captured by a logic is an open problem. If defined too broadly, there are artifical logics which strongly capture PTIME. The right notion is to "effectively strongly capture PTIME", and no logic which does this is known. Proving that there is no such logic implies that PTIME does not equal NPTIME, as the latter is effectively strongly captured by Σ^1_1.

First we give two examples of logics which strongly capture PTIME, but not effectively.

Fix a vocabulary τ and let $\tau_<$ denote the vocabulary obtained by adding the new binary relation symbol $<$. A sentence φ in $\tau_<$ is *order-invariant* iff for all finite τ-structures $\mathcal{A}$ and any two orderings $<_1, <_2$ on A:

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

This notion is not effective: Satisfiability for finite structures is not recursive and a sentence ψ is unsatisfiable in finite structures iff the sentence

$$\psi' \equiv (\psi \to P(\min))$$

is order-invariant (on structures with more than one element), where P is a new unary relation symbol.

Consider now the logic $\mathcal{L}_1$ whose sentences are those of FO(IFP) in the language $\tau_<$ and where satisfaction for τ-structures is defined by:

$\mathcal{A} \models^{\mathcal{L}_1} \varphi$ iff
φ is order-invariant and there is an ordering $<^A$ such that $(\mathcal{A}, <^A) \models^{\text{IFP}} \varphi$.

Similarly, define the logic $\mathcal{L}_2$ by taking as sentences those sentences in the language $\tau_<$ which are order-invariant, with satisfaction for τ-structures defined by:

$\mathcal{A} \models^{\mathcal{L}_2} \varphi$ iff
there is an ordering $<^A$ such that $(\mathcal{A}, <^A) \models^{\text{IFP}} \varphi$.

As FO(IFP) captures PTIME on ordered structures, it follows that the above logics each strongly capture PTIME. But the first logic has an undecidable satisfaction relation and the second logic an undecidable set of sentences. To rule out these examples, we make the following definition, where by a *logic* we mean an assigment to each vocabulary τ a collection of τ-sentences together with a satisfaction relation for these sentences in finite τ-structures.

Definition 18 *The logic $\mathcal{L}$* effectively strongly captures the complexity class *$\mathcal{C}$, written $\mathcal{L} \equiv_{es} \mathcal{C}$, iff:*

1. $\mathcal{L}$ strongly captures $\mathcal{C}$.
2. For every vocabulary τ:
a. The set of τ-sentences is decidable.
b. There is an effective procedure that assigns to every τ-sentence φ a pair (M, F) where M is a Turing machine that accepts $Mod(\varphi)_<$ and f is a code for a function witnessing that M is resource-bounded according to the complexity class $\mathcal{C}$.

Using the fact that $\Sigma_1^1 \equiv_{es}$ NPTIME and a complete problem for NPTIME, it is easy to see that if PTIME = NPTIME, then $\Sigma_1^1 \equiv_{es}$ PTIME.

13.Vorlesung

We could get a good logic for PTIME if we could "canonically" choose an ordering of each structure in polynomial time. This idea is captured by the next definition.

Definition 19 *For a finite relational vocabulary τ let Str$[\tau]$ denote the class of all finite τ-structures. A PTIME* canonization $\mathcal{C}$ *consists of functions C_τ : Str$[\tau] \to$ Str$[\tau_<]$ for each τ such that:*

(1) For all $\mathcal{A}$ in Str$[\tau]$, $<^{C_\tau(\mathcal{A})}$ is an ordering of A and $\mathcal{A}$ is isomorphic to the restriction to τ of $C_\tau(\mathcal{A})$.
(2) If $\mathcal{A}$ is isomorphic to $\mathcal{B}$ then $C_\tau(\mathcal{A})$ is isomorphic to $C_\tau(\mathcal{B})$.
(3) There is a PTIME algorithm that when applied to $(\mathcal{A}, <^A)$ in Str$[\tau_<]$ prouduces the structure $C_\tau(\mathcal{A})$.

Proposition 20 *If there exists a PTIME canonization then there is a logic effectively strongly capturing PTIME.*

Proof. Let $\mathcal{C}$ be a PTIME canonization. Consider the logic $\mathcal{L}$ whose sentences in vocabulary τ are the FO(IFP)$[\tau_<]$ sentences with satisfaction defined by:

$$\mathcal{A} \models^{\mathcal{L},\tau} \varphi \text{ iff } C_\tau(\mathcal{A}) \models^{\text{IFP}} \varphi.$$

We claim that $\mathcal{L}$ effectively strongly captures PTIME. Clearly the set of $\mathcal{L}[\tau]$ sentences is decidable. By earlier work we know that there is an algorithm assigning to each FO(IFP)$[\tau_<]$ sentence φ a pair (M_0, d_0) where M_0 is an x^{d_0} time-bounded TM accepting the class of ordered models of φ. Using a PTIME algorithm for C_τ, we can effectively assign to every $\mathcal{L}[\tau]$ sentence φ a pair (M, d) where M is an x^d time-bounded TM accepting the class $\text{Mod}^{\mathcal{L}}(\varphi)_<$. So $\text{Mod}^{\mathcal{L}}_<$ is in PTIME. Conversely, if K is a class of τ-structures with $K_<$ in PTIME then $K_<$ is $\text{Mod}^{\text{FO(IFP)}}(\varphi)$ for some FO(IFP)$[\tau_<]$ sentence φ, implying that K equals $\text{Mod}^{\mathcal{L}}(\varphi)$. □

For an ordered structure $\mathcal{A}$ let $\mathcal{A}^+$ denote the isomorphic copy of $\mathcal{A}$ in which $<^A$ is the standard order on the numbers less than $|A|$. Then using $\mathcal{A}^+$ any canonization can be improved to obey the stronger property:

$$\mathcal{A} \simeq \mathcal{B} \text{ iff } C_\tau(\mathcal{A}) = C_\tau(\mathcal{B})$$

for τ-structures $\mathcal{A}$, $\mathcal{B}$. As structures of the form $\mathcal{A}^+$ can be canonically coded by words over the alphabet $\{0,1\}$, canonizations in fact lead to "invariantizations", i.e., functions reducing isomorphism to equality on words in the following sense (note the analogy with the notion of "smooth equivalence relation" from descriptive set theory).

Definition 21 *A PTIME* invariantization $\mathcal{I}$ *consists of functions* $I_\tau : Str[\tau] \to \{0,1\}^*$ *such that for each* τ *we have:*

(1) For τ*-structures* $\mathcal{A}$, $\mathcal{B}$, $\mathcal{A} \simeq \mathcal{B}$ *iff* $I_\tau(\mathcal{A}) = I_\tau(\mathcal{B})$.
(2) I_τ *is PTIME computable: there is a PTIME algorithm that produces the word* $I_\tau(\mathcal{A})$ *from a structure* $(\mathcal{A}, <^A) \in Str[\tau_<]$.

14.-15.Vorlesungen

In fact PTIME invariantizations and canonizations are equivalent notions:

Theorem 22 *If there is a PTIME invariantization then there is a PTIME canonization.*

Proof. Suppose that $\mathcal{I}$ is a PTIME invariantization. Let τ be a vocabulary and $\sigma = \tau_<$. We first use $\mathcal{I}_\sigma$ to define for every τ-structure $\mathcal{A}$ with ordering $\prec^A$ a new ordering $<^A$ as follows. If $a_1, \ldots, a_l$ are distinct elements of A let $[a_1, \ldots, a_l]$ be $\{(a_i, a_j) \mid 1 \le i < j \le l\}$ if $l \ge 2$ and otherwise $[a_1] = \{(a_1, a_1)\}$.

Let w_1 be the first element in the lex order on $\{0,1\}^*$ of the set $\{I_\sigma((\mathcal{A}, [a])) \mid a \in A\}$. And choose a_1 to be $\prec^A$-least so that $w_1 = I_\sigma((\mathcal{A}, [a]))$.

Let w_2 be the first element of $\{I_\sigma((\mathcal{A}, [a_1, a])) \mid a \in A,\ a \ne a_1\}$. Let a_2 be $\prec^A$-least so that $w_2 = I_\sigma((\mathcal{A}, [a_1, a_2]))$ and $a_2 \ne a_1$.

After n steps we obtain $a_1, \ldots, a_n$ with $A = \{a_1, \ldots, a_n\}$ and set $<^A = [a_1, \ldots, a_n]$. The desired canonization is the structure $C_\tau^+(\mathcal{A})$ with the standard order on $\{1, \ldots, |A| - 1\}$ which is isomorphic to $C_\tau(\mathcal{A}) = (\mathcal{A}, <^A)$. We must check that if $\mathcal{B}$ is isomorphic to $\mathcal{A}$ then $C_\tau(\mathcal{A})$ equals $C_\tau(\mathcal{B})$. Let $\prec^B$ be an ordering of B and $<^B$ the ordering given by the above procedure applied to $(\mathcal{B}, \prec^B)$. It suffices to show that for all $l \le n$:

$$(*)\ (\mathcal{A}, [a_1, \ldots, a_l]) \simeq (\mathcal{B}, [b_1, \ldots, b_l]).$$

Then for $l = n$ we get $(\mathcal{A}, <^A)$ isomorphic to $(\mathcal{B}, <^B)$ and hence $C_\tau(\mathcal{A}) = C_\tau(\mathcal{B})$. $(*)$ is proved by induction on l. To illustrate, suppose $l = 1$. Suppose that π is an isomorphism of $\mathcal{A}$ onto $\mathcal{B}$. Then for all $a \in A$, π is an isomorphism of $(\mathcal{A}, [a])$ onto $(\mathcal{B}, [\pi(a)])$, so as I_σ is an invariantization,

$$\{I_\sigma((\mathcal{A}, [a])) \mid a \in A\} = \{I_\sigma((\mathcal{B}, [b])) \mid b \in B\}$$

and hence $I_\sigma((\mathcal{A}, [a_1])) = I_\sigma((\mathcal{B}, [b_1]))$. So we have $(\mathcal{A}, [a_1]) \simeq (\mathcal{B}, [b_1])$. $\square$

The question of the existence of a logic which effectively strongly captures PTIME has a nice equivalent formulation in terms of effective enumerations of complexity classes.

Definition 23 *An* effective enumeration $\mathcal{F}$ *of PTIME consists of uniformly computable functions F_τ, τ a finite relational vocabulary, each with domain the natural numbers, such that:*

(1) For any i, $F_\tau(i)$ is a pair (M, d) where M is an x^d time-bounded Turing Machine accepting a class $K_<$ with K consisting of τ-structures.
(2) For any class K of τ-structures such that $K_<$ is in PTIME, there is an i such that $F_\tau(i) = (M, d)$ where M accepts $K_<$.

Proposition 24 *The following are equivalent:*
(a) There is a logic effectively strongly capturing PTIME.
(b) There is an effective enumeration of PTIME.

Proof. (a) implies (b) follows immediately from the definition of effective strong capturing, as if $\mathcal{L}$ effectively strongly captures PTIME then we can effectively enumerate the $\mathcal{L}[\tau]$ sentences and pass to a pair (M, f) where M accepts $\mathrm{Mod}(\varphi)_<$. Conversely, let $\mathcal{F}$ be an effective enumeration of PTIME. First assume that for each τ the range of F_τ is decidable. Then define the logic $\mathcal{L}$ as follows: The set of $\mathcal{L}[\tau]$ sentences is the range of F_τ. And if $F_\tau(i) = (M, d)$ where M accepts the class $K_<$ set:

$$\mathcal{A} \models^{\mathcal{L}} F_\tau(i) \text{ iff } \mathcal{A} \in K.$$

This works. In the general case we use "padding" to replace $\mathcal{F}$ by another enumeration $\mathcal{F}^*$ with the property that the range of $\mathcal{F}^*_\tau$ is decidable for each τ: If $F_\tau(i)$ equals (M, d) then replace M by M^* where the code for the machine M^* is greater than i. (For example, this can be done by adding i-many new, unused instructions to M.) Then to test if (M', d') belongs to the range of F_τ^* we need only look at F_τ restricted to the first code(M') numbers. $\Box$

As mentioned, it is not known if there is a logic which effectively strongly captures PTIME. However there is such a logic if we restrict ourselves to the special class TREE, consisting of those connected directed graphs (A, E^A) where each node has at most one E^A-predecessor. A logic $\mathcal{L}$ *effectively strongly captures PTIME on TREE* iff:

(1) The set of $\mathcal{L}$ sentences is decidable.
(2) A class K of trees is axiomatized by a sentence of $\mathcal{L}$ iff $K_<$ belongs to PTIME.
(3) There is an algorithm that assigns to each sentence φ of $\mathcal{L}$ a pair (M, d) such that M is x^d time-bounded and accepts $\mathrm{Mod}(\varphi)_<$.

The logic that effectively strongly captures PTIME on TREE is *fixed-point logic with counting*, denoted FO(IFP, #). This logic enhances FO(IFP) by allowing terms $\#_x\varphi$, where φ is a formula and x a variable, which is intended to denote the number of x satisfying φ. More precisely:

Fix a vocabulary τ disjoint from the ordering-vocabulary $\{<, S, \min, \max\}$. We consider a language with two sorts. A structure for this language consists of a τ-structure $\mathcal{A}$ together with the disjoint structure with universe $\{0, 1, \ldots, |A| - 1\}$ with its natural interpretation of $\{<, S, \min, \max\}$. We have *point variables* $x, y, z, \ldots$ ranging over the universe A of $\mathcal{A}$ and *number variables* $\mu, \nu, \ldots$ ranging over $\{0, 1, \ldots, |A| - 1\}$. Inductively define:

(a) The *terms of the first sort* are the terms of the vocabulary τ.
(b) The *terms of the second sort* include (but are not restricted to) the terms of the vocabulary $\{<, S, \min, \max\}$.
(c) Atomic formulas in either sort are formulas.
(d) If X is a second-order variable of arity (n_1, n_2) (i.e., of first-sort arity n_1 and second-sort arity n_2) then $X\bar{t}\bar{\rho}$ is a formula where $\bar{t}$ is a length n_1 sequence of terms of the first sort and $\bar{\rho}$ is a length n_2 sequence of terms of the second sort.
(e) Formulas are closed under logical connectives and first-order quantifiers over both sorts.
(f) If φ is a formula then so is $[\mathrm{IFP}_{\bar{x}\bar{u}X}\varphi]\bar{t}\bar{\rho}$ (where $\bar{x}, \bar{u}, \bar{t}, \bar{\rho}$ have the right

arities).
(g) if φ is a formula then $\#_x\varphi$ is a term of the second sort.

The interpretation of $\#_x\varphi(x,\bar{a})$ in $\mathcal{A}$ is the number of $a \in A$ such that $\varphi[a,\bar{a}]$ holds in $\mathcal{A}$.

We want to show that FO(IFP, #) effectively strongly captures PTIME on TREE. For this the following lemma is useful.

Lemma 25 *Suppose that there is a formula $\varphi(\mu,\nu)$ of FO(IFP, #) such that for all trees $\mathcal{A}$:*

$$\mathcal{A} \simeq (\{0,1,\ldots,|A|-1\}, \varphi^{\mathcal{A}}(\cdot,\cdot)).$$

Then FO(IFP, #) effectively strongly captures PTIME on TREE.

Remark. In the above isomorphism: on the left, $\mathcal{A}$ is to be viewed as a tree without the second sort and on the right as a tree with the second sort.

Proof sketch. Suppose that K is a class of trees and $K_<$ is in PTIME. Then $K_< = \mathrm{Mod}(\psi)$ for some sentence ψ in FO(IFP) in the vocabulary with $<$. Using the hypothesis of the lemma, we can translate ψ into a sentence ψ^* of FO(IFP, #) replacing $<$ with the ordering of the second sort and the edge relation with the relation defined by φ. So K is axiomatizable in FO(IFP, #). The converse holds as the logic FO(IFP, #) with $<$ can only axiomatize PTIME classes of structures. $\Box$

Now we prove the hypothesis of the lemma. Let $\mathcal{A}$ be a tree. By induction from leaves to root, we define for each $u \in A$ a copy of the subtree $\mathcal{A}_u$ with root u on an initial segment of $\{0,1,\ldots,|A|-1\}$, the number part of $\mathcal{A}$. The tree relation will be given by $Xu\cdot\cdot$ where X is a ternary relation with first component for point variables and second and third components for number variables. The case of a leaf u is trivial. Suppose that u has successors $v_1,\ldots,v_l$. By induction the trees $\mathcal{A}_{v_1},\ldots,\mathcal{A}_{v_l}$ have isomorphic copies on initial segments of the number part of $\mathcal{A}$ given by $Xv_1\cdot\cdot$, $Xv_2\cdot\cdot$, $\cdots$, $Xv_l\cdot\cdot$. Using the ordering these copies can be ordered lexicographically in a first-order definable way as follows:

$Xv_i\cdot\cdot \prec Xv_j\cdot\cdot$ iff
"$Xv_i\cdot\cdot \neq Xv_j\cdot\cdot$ and the lexicographically least pair $(\mu\nu)$ where they differ belongs to $Xv_i\cdot\cdot$".

Abbreviate the above by $v_i \prec v_j$ and if neither $v_i \prec v_j$ nor $v_j \prec v_i$ then we write $v_i \equiv v_j$. Now we illustrate the definition of the copy of $\mathcal{A}_u$ on an initial segment of $\{0, 1, \ldots, |A| - 1\}$ via an example: Suppose that $l = 4$ and $v_1 \prec v_2 \equiv v_3 \prec v_4$. Then we assign 0 to u and follow it with a copy of $Xv_1 \cdot\cdot$, two copies of $Xv_2 \cdot\cdot$ and one copy of $Xv_4 \cdot\cdot$. To inductively define this copy of $\mathcal{A}_u$ (and ultimately of $\mathcal{A} = \mathcal{A}_{u_0}$ where u_0 is the root of $\mathcal{A}$) we need IFP together with a function that keeps track of the total number of elements of subtrees $\mathcal{A}_{v'}$ for successors v' of u that are below a given successor v of u in $\prec$; this function is

$$\delta(u, v) = \#\{w \mid \exists v'(Euv' \wedge v' \prec v \wedge w \in \mathcal{A}_{v'})\},$$

where E denotes the edge relation. $\Box$