Toward logic tailored for computational complexity

by Yuri Gurevich[1]

Computer Science
The University of Michigan
Ann Arbor, Michigan 48109

Abstract. Whereas first-order logic was developed to confront the infinite it is often used in computer science in such a way that infinite models are meaningless. We discuss the first-order theory of finite structures and alternatives to first-order logic, especially polynomial time logic.

Introduction

Turning to theoretical computer science a logician discovers with pleasure an important role of first-order logic. One of the fashionable programming languages – PROLOG – is based on first-order logic; variants of first-order logic – Tuple Calculus, Relational Algebra, Domain Calculus – are used as query languages to retrieve information from relational databases; et cetera.

The database applications of first-order logic are of special interest to us here. In this connection let us mention that relational databases are not a side issue in the data field. The relational data model together with the network and the hierarchical data models are "the three most important 'data models', the models that have been used in the great bulk of commercial database systems" [Ul, Section 1.4]. The relational data model brought a Turing Award to its inventor E.F. Codd. The three query languages, mentioned above, were also introduced by Codd and are important: "A language that can (at least) simulate tuple calculus, or equivalently, relational algebra or domain calculus, is said to be complete" [Ul, Section 6.1].

Some of the new applications of first-order logic are unusual in that only finite structures are of interest. In particular, relational databases can be seen as finite first-order structures (for the purpose of this paper), and the query languages, mentioned above, express exactly the first-order properties of relational databases. The question arises how good is first-order logic in handling finite

[1]Supported in part by NSF grant MCS83-01022

structures. It was not designed to deal exclusively with finite structures. In a sense the contrary is true. It was developed as a tool in Foundations of Mathematics, especially when mathematicians and philosophers confronted paradoxes of the Infinite.

We do not question here the greatness of first-order logic of not necessarily finite structures. Taking into account how elegant, natural and expressive first-order logic is, it is actually amazing that formulas true in *all* structures (of an appropriate vocabulary) are exactly the ones for which there *exist* proofs in a specific formal system. (Let us also recall the unique character of first-order logic [Lin].) But what happens to recursive axiomatizability, compactness and other famous theorems about first-order logic in the case of finite structures? We address this question in §1. Our feelings about the answer are expressed in the title of §1: Failure of first-order logic in the case of finite structures.

In §2 we address a certain ineffectiveness of famous theorems about first-order logic. Consider for example Craig's Interpolation Theorem: for each valid implication $\varphi\rightarrow\psi$ there is an interpolant θ such that

$$\text{vocabulary}(\theta) \subseteq \text{vocabulary}(\varphi) \cap \text{vocabulary}(\psi)$$

and the implications $\varphi\rightarrow\theta$ and $\theta\rightarrow\psi$ are valid. No total recursive function constructs an interpolant from the given implication [Kr]. There is no recursive bound on the size of the desired interpolant in terms of the size of the given implication [Fr]. Moreover, weaken the interpolation theorem by replacing "the implications $\varphi\rightarrow\theta$ and $\theta\rightarrow\psi$ are valid" by "the implications $\varphi\rightarrow\theta$ and $\theta\rightarrow\psi$ are valid in all finite structures of appropriate vocabularies". Still there is no recursive bound on the size of the desired interpolant in terms of the size of the given implication.

What is the use of criticizing first-order logic if we cannot come up with a reasonable alternative? We think here about applications where one needs at least the expressive power of first-order logic, like PROLOG or relational query languages. "It is the case that almost all modern query languages embed within them on of the three notations" [Ul, Section 6.1]. (The three notations are the tuple calculus, the relational algebra, and the domain calculus.)

One would like to enrich first-order logic so that the enriched logic fits better the case of finite structures. The first temptation of a logician would be to regain recursive axiomatizability. But no extension of the first-order theory of finite structures is recursively axiomatizable. (Satisfiability of first-order formulas on finite structures is recursively axiomatizable. But this axiomatizability provides only a criterion of *existence* of a formal proof for *existence* of a finite model. It is not interesting. The whole point of axiomatizability was to provide an existential criterion for a universal statement.)

Another temptation is to consider second-order logic (without third-order predicates or functions) or its fragments (like existential second-order logic) as an alternative to first-order logic. Confining ourselves to finite structures, we consider this alternative in §3.

Second-order logic is certainly elegant, natural and much more expressive than first-order logic. Second-order logic itself becomes more attractive in the case of finite structures: no nonstandard models, no distinction between the weak and the strong versions of second-order logic, etc. There is however one important - from the point of view of computer science – property of first-order logic that is lost in the transition to second-order logic. For every first-order sentence φ there is an algorithm that, given a presentation of a structure S of the vocabulary of φ, computes the truth-value of φ on S within time bounded by a polynomial in the cardinality $|S|$ of S (and within working space bounded by $\log |S|$). In other words, first-order properties are PTIME (and LOGSPACE) computable. Second-order properties and even existential second-order properties are not PTIME computable unless $P = NP$. If one takes the popular point of view that feasible computations are PTIME bounded and that P is probably different from NP then second-order logic is not a good alternative to first-order logic.

Let us mention that computer scientists do feel that first-order logic is unreasonably restrictive. PROLOG does have non-first-order features, and it was suggested to augment the essentially first-order query languages by different operators preserving feasible computability of queries. Of course the notion of feasibility varies with applications. From the point of view of PTIME computability, the least fixed point operator LFP [AU] appeared to be especially important. It preserves PTIME computability and has great expressive power.

A natural idea arises to extend first-order logic in such a way that exactly PTIME (LOGSPACE, etc.) computable properties of structures are expressible in the extended logic. Chandra and Harel [CH2] considered the extension FO + LFP of first-order logic by LFP from that point of view and discovered that FO + LFP does not capture PTIME. It turned out, however, that FO + LFP does capture PTIME in the presence of linear order [IM1, Var]. In §4 we discuss fixed points and logics with order (as a logical constant) tailored for PTIME.

In §5 we return to some of the famous theorems about first-order logic and consider whether their analogues hold in the ease of logic specially designed for PTIME. More specifically, we consider the analogues of Craig's Interpolation Theorem, Beth's Definability Theorem and the Weak Beth Definability Theorem for polynomial time logic. These analogues happen to be equivalent to natural complexity principles whose status is unknown.

A lot of interesting problems arise. Design a logic that captures PTIME even in absence of linear order, or prove that there is no reasonable such logic if $P \neq NP$. What is a logic? What is a complexity class? Can every reasonable complexity class be captured by a logic in the presence of linear order? Capture LOGSPACE, NLOGSPACE, LOG^2SPACE, $\text{LOG}^2\text{SPACE} \cap \text{PTIME}$, etc. in the presence of linear order. What are complexity tailored logics good for? Are complexity bounded programming languages useful? Some answers can be found in [Im2] and [Gu3].

Our terminology is more or less standard. We use the term "vocabulary" rather than "signature" or "similarity type", and we use the term "structure" rather than "model" or "algebraic system". Our vocabularies are always finite.

Acknowledgements. I am very grateful to Andreas Blass and Neil Immerman for very useful discussions related to this paper.

§1. Failure of first-order logic in the case of finite structures

We examine famous theorems about first-order logic in the case when only finite structures are allowed. The terms formula and sentence will refer in this section to first-order formulas and first order sentences. As usual, a sentence is a formula without free individual variables.

Recall that a formula φ is called valid (or logically true) if it is true in every structure of the vocabulary of φ, a formula φ (resp. a set Φ of formulas) is said to imply a formula ψ logically if ψ is true in every model of φ (resp. of Φ) whose vocabulary includes that of ψ and formulas φ, ψ are called logically equivalent if each of them logically implies the other. We will say that a formula φ is *valid in the finite case* if it is true in every finite structure of the vocabulary of φ, a formula φ (resp. a set Φ of formulas) *implies* a formula ψ *in the finite case* if ψ is true in every finite model of φ (respectively of Φ) whose vocabulary includes that of ψ, and formulas φ, ψ are *equivalent in the finite case* if each of them implies the other in the finite case.

The Soundness and Completeness Theorem is formulated usually for a specific logical calculus. It states that a formula is valid iff it is provable in the calculus. The calculus-independent meaning of this theorem is that first-order logic is recursively axiomatizable, which boils down to the fact that valid formulas are recursively enumerable. Trakhtenbrot [Tr] proved that the formulas valid in the finite case are not recursively enumerable. Therefore first-order logic is not recursively axiomatizable in the finite case, and the Soundness and Completeness Theorem fails for any logical calculus in the finite case.

Remark. Tiny fragments of first-order logic are not axiomatizable recursively in the case of finite structures. For example, let σ be a vocabulary that consists of one binary predicate symbol. The $\exists^3\forall^*$ σ-sentences (i.e. the prenex σ-sentences with prefixes $\exists^3\forall^n$), that are valid in the finite case, are not enumerable recursively [Gul, Ko]. Summaries of results of that sort can be found in [Gu2]. Goldfarb claims that even $\exists^2\forall^*$ σ-sentences with equality, valid in the finite case, are not enumerable recursively [Go].

The Compactness Theorem for first-order logic states that if a set Φ of formulas logically implies another formula ψ then some finite subset of Φ logically implies ψ. The theorem fails in the finite case. Let for example $\Phi = \{\varphi_n : n>l\}$ where every sentence φ_n states existence of at least *n* different elements, and let ψ be any logically false formula. Then Φ implies ψ in the finite case; however no finite subset of Φ implies ψ in the finite case.

The Craig Interpolation Theorem states that if a formula φ logically implies a formula ψ then there is a formula θ (an interpolant) such that

$$\text{vocabulary}\,(\theta) \subseteq \text{vocabulary}\,(\varphi) \cap \text{vocabulary}\,(\psi)\ ,$$

φ logically implies θ, and θ logically implies ψ.

The interpolation theorem implies the Beth Definability Theorem that states the following. Suppose that a sentence $\varphi(P)$ defines an *l*-ary relation *P* implicitly i.e. if *P*' is a new *l*-ary predicate symbol then $\varphi(P)$ and $\varphi(P')$ imply

$$\forall x_1 \ldots \forall x_l\, (P(x_1, \ldots ,x_l) \leftrightarrow P'(x_1, \ldots ,x_l))\ .$$

Then there is an explicit first-order definition of the same relation i.e. there is a formula $\theta(x_1, \ldots ,x_l)$ such that

$$\text{vocabulary}\,(\theta) \subseteq \text{vocabulary}\,(\varphi(P)) - \{P\}$$

and $\varphi(P)$ logically implies

$$\forall x_1 \ldots \forall x_l\, (P(x_1, \ldots ,x_l) \leftrightarrow \theta(x_1, \ldots ,x_l)).$$

If $\varphi(P)$ and P' are as in the antecedent of the Beth Definability Theorem then $\varphi(P)$ & $P(x_1, \ldots ,x_l)$ logically implies $\varphi(P') \rightarrow P'(x_1, \ldots ,x_l)$, and the corresponding interpolant is the desired explicit definition. The same proof shows that the finite case version of the interpolation theorem implies the finite case version of the definability theorem.

The Weak Definability Theorem is the result of strengthening the antecedent of the Beth Definability Theorem. The antecedent of the Beth Definability Theorem states that for every structure of the vocabulary

$$\sigma = \text{vocabulary}\,(\varphi(P)\,) - \{P\}$$

there is at most one relation *P* that satisfies $\varphi(P)$. The antecedent of the Weak

Definability Theorem states that for every σ-structure there is a unique relation *P* that satisfies φ(P).

Theorem 1. *The Craig Interpolation Theorem, the Beth Definability Theorem and the Weak Definability Theorem fail in the finite case.*

Proof. Let us recall the definition of the quantifier depth of a formula:

q.d. (a quantifier-free formula) = 0

q.d. (a Boolean combination of formulas $\alpha_1, \ldots, \alpha_m$) = $\max\{$q.d.(α_1), . . . , q.d.(α_m)$\}$

q.d. ($\forall x\alpha$) = q.d. ($\exists x\alpha$) = l + q.d. (α).

Lemma.

(i) *Suppose that* α *is a sentence in the vocabulary* {<} *of order, n is the quantifier depth of* α, *and* A, B *are finite linear orders of cardinalities* $|A|, |B| \geq 2^n$. *Then* α *does not distinguish between* A *and* B *i.e.* A *satisfy* α *iff* B *satisfies* α.

(ii) *There is no sentence* α *in the vocabulary* {<} *such that an arbitrary finite linear order* S *satisfies* α *iff the cardinality* |S| *is even.*

(iii) *There is no formula* θ(x) *in the vocabulary* {<} *such that if* S *is a finite order* $a_1 < a_2 < \ldots < a_n$ *then* $\{x: S \models 1= \theta(x)\} = \{a_k: k$ *is even*$\}$.

Proof of Lemma.

(i) We use the Ehrenfeucht games [Eh]. It suffices to exhibit a winning strategy for player II in the Ehrenfeucht game $G_n(A, B)$. Without loss of generality no element is picked twice during the game. The proposed strategy is to ensure the following. Let $a_1 < a_2 < \ldots < a_k$ and $b_1 < b_2 < \ldots < b_k$ be the elements chosen in *A* and *B* respectively during the first *k* steps of the game. Let $A_0, A_1, \ldots, A_k$ be the segments $[\min(A), a_1]$, $[a_1, a_2], \ldots, [a_k, \max(A)]$ of *A*, and let $B_0, B_1, \ldots, B_k$ be the respective segments of *B*. Then for every i = l, . . . , k the elements a_i, b_i were chosen at the same step of the game, and either $|A_i| = |B_i|$ or $|A_i|, |B_i| > 2^{n-k}$ for $0 < i < k$, and either $|A_i| = |B_i|$ or $|A_i|, |B_i| \geq 2^{n-k}$ for $i \in \{0,...,k\}$.

(ii) The statement follows from (i).

(iii) If $\theta(x)$ defines the set of even elements in every finite linear order then the sentence

$$\exists x\ (x \text{ is maximal and } \theta(x))$$

holds in an arbitrary finite linear order *S* iff $|S|$ is even. □

Since the interpolation theorem implies the definability theorem and the definability theorem implies the weak definability theorem, it suffices to refute the weak definability theorem. It is easy, however, to construct separate counterexamples to all three theorems.

Write a sentence α stating that $<$ is a linear order, let *P*, *Q* be distinct unary predicates. Write a sentence $\beta(P)$ in the vocabulary $\{<, P\}$ such that if *S* is a finite linear order $a_1 < a_2 < \ldots < a_n$ and *S* satisfies $\beta(P)$ then $\{x: S \models P(x)\} = \{a_k: k \text{ is even}\}$. (Write that *P* does not contain the first element, and the successor of an element *x* belongs to *P* iff *x* does not belong to *P*.) Obviously α & $\beta(P)$ & $P(x)$ implies $\beta(Q) \rightarrow Q(x)$ in the finite case. If the interpolation theorem were true in the finite case then the interpolant would violate the statement (iii) of the Lemma.

Obviously, α & $\beta(P)$ defines *P* implicitly in finite structures. If the definability theorem were true in the finite case then the explicit definition of *P* would violate the statement (iii) of the Lemma. Finally, the sentence

$$(\alpha \rightarrow \beta(P))\ \&\ (\neg\alpha \rightarrow \neg\exists x P(x))$$

defines *P* uniquely in the finite case. If the weak definability theorem were true in the finite case then the explicit definition would violate the statement (iii) of the Lemma. Theorem 1 is proved.

Remark. The formula $\beta(P)$ in the proof of Theorem 1 can be simplified if we use an individual constant for the first element in the order and an additional binary predicate symbol for the successor relation. The Lemma remains true for the richer vocabulary if 2^n is changed to 2^{n+1} (with an obvious change in the proof).

A sentence φ is said to be preserved under substructures if every substructure of a model of φ is a model of φ. According to the Substructure Preservation Theorem [CK, §3.2], a sentence φ is preserved under substructures iff it is logically equivalent to a universal sentence.

Theorem 2 (Tait) . *The Substructure Preservation Theorem fails in the case of finite structures. In other words, there is a sentence φ such that any substructure of a finite model of φ is a model of φ, yet φ is not equivalent to any universal sentence in the finite case.*

Proof. Let (φ_1 be the universal closure of the conjunction of the following formulas (where $x \le y$ abbreviates $x = y \lor x < y$) :

$$(x < y \mathbin{\&} y < z) \rightarrow x < z,$$

$$\neg (x < x),$$

$$x \le y \lor y \le x,$$

$$0 \le \mathbf{x},$$

$$[S(x, y) \mathbin{\&} y \ne 0] \rightarrow [x < y \mathbin{\&} (z \le x \lor y \le z)],$$

$$S(x, 0) \rightarrow y \le x.$$

φ_1 states that $<$ is a linear order, 0 is the minimal element, and *Sxy* implies that either *y* is the successor of *x* or else *x* is the maximal element and $y = 0$. Let φ_2 be $\forall x \exists y\, S(x, y)$, and let φ be $\varphi_1 \mathbin{\&} (\varphi_2 \rightarrow \exists x\, P(x))$ where *P* is a unary predicate symbol.

First we check that φ is preserved under substructures of finite models. Suppose that *A* is a finite model of φ and *B* is a substructure of *A*. Then *B* contains 0 and satisfies φ_1. If *B* does not satisfy φ_2, then it satisfies the second conjunct of φ by default. If *B* satisfies φ_2 then $B = A$ and *B* satisfies φ.

Next, let α be a sentence $\forall x_1 \ldots \forall x_n\, \beta(x_1, \ldots, x_n)$ where β is quantifier-free. Let *A* be a model of $\varphi_1 \mathbin{\&} \varphi_2$ such that the vocabulary of *A* includes that of α,*A* has at least n+2 elements and *P* is empty in *A*, so that φ fails in A. If α is true in *A* then it is not equivalent to φ in the finite case. Suppose that α is false in *A*. Then $\beta(c_1, \ldots, c_n)$ is false in *A* for some $c_1, \ldots, c_n$. Choose $d \in A$ different from $0, c_1, \ldots, c_n$, and put *d* into *P*. The resulting structure *B* satisfies φ. However $\beta(c_1, \ldots, c_n)$ remains false in *B*. Hence α is false in *B*, and α is not equivalent to φ in the finite case. □

I did not perform an exhaustive study of important theorems about first-order logic in the finite case. Some theorems become meaningless in the finite case.

Some theorems do survive: the game criterion for two structures to be indistinguishable by sentences of a given quantifier depth [Eh], composition theorems of the sort found in [FV], etc. Moreover, some theorems were specifically proved for the finite case: the 0-1 Law Theorem for example [GKLT, Fa]. Too often however we see the familiar pattern: the proof uses a kind of compactness argument and the theorem fails in the finite case. Sometimes a weaker version of the theorem in question survives. Here is an example. Recall that an $\forall^*\exists^*$ sentence is a prenex sentence with a prefix $\forall^m\exists^n$.

Theorem 3 (Compton). *Let φ be an $\forall^*\exists^*$ sentence without function symbols. If φ is preserved by substructures of its finite models then it is equivalent to some universal sentence in the finite case.*

Proof. First let us recall a relativized version of the Substructure Preservation Theorem:

Let T_0 be a first-order theory, and α be a sentence in the language of T_0. Suppose that for every model *A* of T_0 and for every substructure *B* of *A* that is a model of T_0, if *A* is a model of α then *B* is a model of α. Then α is equivalent in T_0 to some universal sentence.

The usual proof of the Substructure Preservation Theorem is easily relativizable: just take Δ to be the set of all sentences, that are equivalent in T_0 to universal sentences, in the proof of Theorem 3.2.2 in [CK].

In our application T_0 is the first-order theory of finite structures of the vocabulary of φ. Let *A* be a (possibly infinite) model of T_0, that satisfies φ. Let *B* be a substructure of *A* that is also a model of T_0. It suffices to prove that *B* satisfies φ.

First, we show that an arbitrary finite substructure A_0 of *A* satisfies φ. Write an existential sentence α stating existence of elements that form a structure isomorphic to A_0. The sentence $\alpha \,\&\, \varphi$ has a finite model A_1: otherwise $T_0 \models \neg(\alpha \,\&\, \varphi)$ which contradicts the fact that *A* is a model of T_0, α and φ. Since A_1 satisfies α it has a substructure isomorphic to A_0. Now use the fact that φ holds in A_1 and is preserved by substructures of finite structures.

Recall that φ is

$$\forall x_1 \dots \forall x_m \exists y_1 \dots \exists y_n\ \psi(x_1, \dots, x_m, y_1, \dots, y_n)$$

for some quantifier-free formula ψ. We argue by reduction ad absurdum. Suppose that *B* fails to satisfy φ. Then there are elements $a_1, \dots, a_m$ such that the universal formula

$$\forall y_1 \dots \forall y_n\ \neg\psi(a_1, \dots, a_m, y_1, \dots, y_n)$$

holds in *B*. This universal formula logically implies $\neg\varphi$ and holds in the substructure $A_0 = \{ a_1, \dots, a_m\}$ of *B* (because universal formulas are preserved by substructure) . Thus a finite substructure of *A* fails to satisfy φ which is impossible. □

Note that the counterexample to the Substructure Preservation Theorem, constructed in the proof of Theorem 2 is logically equivalent to an $\exists^*\forall^*$ sentence. Thus Theorems 2 and 3 delimit each other.

Historical Remarks. I am not the first to discover that Craig's Interpolation Theorem and Beth's Definability Theorem fail in the finite case. (A question of Steve Simpson led me from Craig's Theorem to Beth's Theorem.) Ron Fagin knew about the failure. It was probably discovered long ago though I do not have any reference.

Theorem 2 was proved in [Ta]. The proof above is due to Gurevich and Shelah (that were not aware [Ta]). Theorem 3 was formulated and proved by Kevin Compton in a letter [Co] to me.

§2. An ineffective side of first-order logic

We saw in §1 that Craig's Interpolation Theorem, Beth's Definability Theorem, the Weak Definability Theorem and the Substructure Preservation Theorem fail in the case of finite structures. One may be tempted to allow infinite structures (to allow infinite relational databases in database theory) in order to regain these wonderful theorems; see [Va] for example. There is however a catch there. Let us speak, for example, about the weak definability theorem. Even if you happen to know that $\Phi(P)$ implicitly defines a relation *P* in every – finite or infinite – structure and even if you are interested in an explicit definition of the same relation *P* in finite structures only, still constructing the desired explicit definition from the given implicitly definition may be most problematic. This is the point of the present section. Again, the terms formula and sentence mean first-order formulas and first-order sentences. The length of a formula φ is denoted $|\varphi|$.

So then, how constructive are the wonderful theorems mentioned above? In a certain sense the interpolation theorem is very constructive. The desired interpolant for a valid implication $(\varphi \rightarrow \psi)$ is easily constructible from a *proof* of $(\varphi \rightarrow \psi)$ in an appropriate predicate calculus [Cr]. In the same sense the definability theorem is very constructive because the desired explicit definition can be found as an interpolant for an implication that is easily built from the given implicit definition, see §1.

There are also partial recursive functions *f* and *g* such that if $(\varphi \rightarrow \psi)$ is a valid implication then $f(\varphi \rightarrow \psi)$ is an interpolant for $(\varphi \rightarrow \psi)$, and if $\varphi(P)$ is an implicit definition of a relation *P* then $g(\varphi(P))$ is an explicit definition of the same relation. However, there are no total recursive functions *f* and *g* with the same properties [Kr]. Moreover, there are no total recursive functions that bound the length of the desired interpolant or explicit definition in terms of the length of a given formula [Fr]. Even the weak definability theorem is ineffective in that sense: the length of the desired explicit definition is not bounded by any recursive function of the length of a given implicit definition. The next theorem gives a straightforward proof of this result of Friedman and strengthens it in a way related to finite structures.

Theorem 1. *For every total recursive function* f *there is a sentence* $\varphi(P)$ *such that*

(i) $\varphi(P)$ *implicitly defines a relation* P *in every structure of the vocabulary* σ = *vocabulary* $(\varphi(P)) - \{P\}$, *and*

(ii) *if* ψ *is an explicit definition of the same relation* P *in every finite* σ*-structure, then* $|\psi| \geq f(|\varphi(P)|)$.

Proof. Given a total recursive function *f* we construct an auxiliary total recursive function *g*. The exact definition of *g* will be given later. Let *M* be a Turing machine that computes *g*. We suppose the following about *M*. Its internal states are $q_0, \ldots, q_m$ here q_0 is the initial state and q_m is the halting state. The only tape of *M* is one-way infinite, the tape alphabet is {0, 1} where 0 is also the blank. An instruction of *M* is a 5-tuple $q_i a q_j b d$ where $d \in \{-l, 0, l\}$ indicates whether the head of *M* will move to the left, stay still or move to the right. If at moment 0 the state of *M* is q_0, the head is in cell 0 and the tape word is 1^n then *M* will eventually halt in the halting state q_m with the tape word $1^{g(n)}$.

In order to describe computations of *M* by formulas we introduce unary predicates $q_0(t), \ldots, q_m(t)$ to indicate the state at moment *t*, a binary predicate $H(x, t)$ to indicate that the head is in cell *x* at moment *t*, a binary predicate $C(x, t)$ to indicate that the content of cell *x* at moment *t* is 1, and unary predicates $D_{-1}(t)$, $D_0(t)$, $D_{+1}(t)$ to indicate the move of the head that the machine is instructed at moment t to perform.

In order to use all these predicates properly, we need binary predicates <, S and an individual constant 0. Let a sentence φ_0 state that < is a linear order, 0 is the minimal element, *S* is the corresponding successor relation, and every nonmaximal element has a successor.

A sentence φ_1^n describes the initial configuration of *M* with the input 1^n. It is the conjunction of sentences

$q_0(0)$, $H(0,0)$,

$\exists x_0 \ldots \exists x_n\, [x_0 = 0$ and $\wedge_{i<n} S(x_i, x_{i+1})$ and $\wedge_{i<n} C(x_i, 0)$ and $\neg C(x_n, 0)]$,

$\forall x \forall y\, [\neg C(x, 0)$ and $x \leq y$ imply $\neg C(y, 0)]$.

A sentence φ_2 describes one computational step. It is the universal closure of a quantifier-free conjunction. Every instruction $q_i a q_j b d$ contributes the conjunct

$$[q_i(t) \ \&\ H(x, t) \ \&\ C_a(x, t) \ \&\ S(t, t') \text{ implies } q_j(t') \ \underline{\&}\ C_b(x, t') \ \&\ D_d(t)]$$

where C_1, C_0 are C, $\neg$C respectively. In addition the quantifier-free part of φ_2 has the following conjuncts:

$[\neg q_i(t) \text{ or } \neg q_i(t)]$ for $0 < I < j \leq m$,

$[H(x, t) \text{ and } H(y, t) \text{ imply } x = y]$,

$[\neg D_d(t) \text{ or } \neg D_e(t)]$ for $-l \leq d < e \leq l$,

$[D_0(t) \ \&\ H(x, t) \ \&\ S(t, t') \text{ implies } H(x, t')]$,

$[D_1(t) \ \&\ H(x, t) \ \&\ S(t, t') \ \&\ S(x, x') \text{ implies } H(x', t')]$,

$[D\text{-}_1(t) \ \&\ S(t, t') \ \&\ S(x, x') \ \&\ H(x', t) \text{ implies } H(x, t')]$,

$[\neg H(x, t) \text{ and } S(t, t') \text{ imply } (C(x, t') \leftrightarrow C(x, t))]$.

A sentence φ_3 describes what happens after halting. It is the universal closure of the formula

$$[q_m(t) \text{ and } t < u \text{ imply } (\wedge_{i \leq m} \neg q_i(u) \text{ and } \neg H(x, u)$$
$$\text{and } \neg C(x, u) \text{ and } (\wedge_{-1 \leq d \leq 1} \neg D_d(u))] .$$

<u>Lemma</u>. *For every model* A *of* φ_0 *and for every natural number* n *there are unique predicates*

$$q_0, \ldots, q_m, H, C, D_{-1}, D_0, D_1$$

on *A* that satisfy φ_1^n & φ_2 & φ_3.

<u>Proof</u> is straightforward. In particular, the sentences φ_0, φ_1^n, φ_2 and φ_3 imply that for every *t* there is a unique *x* with H(x, t): the head does not slip from the tape because *M* computes a total function, and if $D_1(t)$, H(x, t), S(t, t') hold then $x \leq t < t'$ and there is *x*' such that S(x, x'), H(x', t') hold.

Let *P* be a ternary predicate symbol. Write a sentence $\varphi^n(P)$ that states the following. If φ_0 fails or there are at most *m*+3 elements then *P* is empty. If φ holds and there are more than *m*+3 elements then

(a) φ_1^n, φ_2, φ_3 hold where $q_i(t)$, $D_d(t)$, H(x, t), C(x, t) abbreviate P(0, i, t), P(0, m+2+d, t), P(l, x, t), P(2, x, t) respectively, and

(b) P(0, x, t) fails for x > m+3, and P(x, y, t) fails for x > 2.

When numbers 1, 2, etc. appear as arguments of *P* they mean of course the successor of 0, the successor of the successor of 0, etc. It is easy to see that φ^n implicitly defines a relation *P* in every structure of the vocabulary $\sigma = \{<, 0, S\}$. Let ψ^n be an explicit definition of the same relation in every finite σ-structure.

Note that φ_1^n and the quantifier depth of φ^n do not depend on the choice of *g*. Define g(n) to be the power of 2 such that

$$\log_2 g(n) = f(|\varphi_1^n| + n) + \text{q.d.}(\varphi^n) + 1.$$

φ_1^n is the only part of φ^n that depends on *n*. It occurs in φ^n only once. Thus the number $k = |\varphi^n| - |\varphi_1^n|$ does not depend on *n*. Let $\varphi = \varphi^k$ and $\varphi = \psi^k$. Then

$$\log_2 g(k) = f(|\varphi|) + \text{q.d.}(\varphi) + 1.$$

Let α be the sentence

$$[\varphi(\psi) \text{ and } \exists t q_m(t)].$$

Every model of α reflects the whole computation *M* and has at least g(k) elements. By the Remark following the Lemma in §1, $g(k) \leq 2^{1 + \text{q.d.}(\alpha)}$. Hence

$$\text{q.d.}(\alpha) + 1 \geq \log_2 g(k) = f(|\varphi|) + \text{q.d.}(\varphi) + 1$$

But $\text{q.d.}(\alpha) \leq \text{q.d.}(\varphi) + \text{q.d.}(\psi)$. Hence $|\psi| \geq \text{q.d.}(\psi) \geq f(|\psi|)$. □

Remark 1. It is easy to make the relation *P* of Theorem 1 unary. The idea is to use auxiliary elements to code triples of real elements.

Remark 2. Mundici exhibits in [Mu] short valid implications $(\varphi \rightarrow \psi)$ whose interpolants are enormously long. The proof of Theorem 1 can be used for analogous purposes.

Theorem 2. *For every total recursive function* f *there is a sentence* φ *such that*

(i) φ *is preserved by substructures, and*

(ii) *if* ψ *is a universal sentence that is equivalent to* φ *in every finite structure of the vocabulary of* φ *then* $|\psi| \geq f(|\varphi|)$. □

Proof. Let *f* be a total recursive function. As in the proof of Theorem 1, let *g* be an auxiliary total recursive function (specified later) and let *M* be a Turing machine that computes *g*. Once again we describe computations of *M* by first-order sentences. However, we take some additional care to make the desired description preserved under substructures.

Instead of the sentence φ_0 in the proof of Theorem 1 we use sentences φ_1, φ_2 from the proof of Theorem 2 in §1. We call them φ_{01} and φ_{02} here. We split the sentence φ_1^n from the proof of Theorem 1 into a conjunction $\varphi_{11} \& \varphi_{12}^n$ where φ_{12}^n is the existential conjunct of $\varphi^n{}_1$ and φ_{11} is the conjunction of the two other conjuncts of φ_1^n. Let φ^n be the sentence

$$\varphi_{01} \ \& \ \varphi_{11} \& \varphi_2 \& \varphi_3 \ \& \ [\varphi_{02} \& \varphi_{12}^n \rightarrow \exists t \ \exists x \ (q_m(t) \ \& \ x \leq t \ \& \ Q(x)]$$

where φ_2 and φ_3 are the sentences from the proof of theorem 1 and Q is a new unary predicate.

First we check that every φ^n is preserved by substructures. Let *A* be a model of φ^n. Every substructure *B* of *A* contains 0 and satisfies the sentences $\varphi_{01}, \varphi_{11}, \varphi_2, \varphi_3$ because universal sentences are preserved by substructures. If *B* does not satisfy φ_{02} or φ_{12}^n then it satisfies the last conjunct of φ^n by default. Suppose that φ^n satisfies φ_{02} and φ_{12}^n. Since *B* satisfies φ_{02} it is closed in *A* under successors. If *A* is finite then *B* is equal to *A* and satisfies φ^n. Suppose *A* is infinite. Then *B* includes the least substructure of *A* closed under successors whose elements can be identified with natural numbers in the obvious way. Since *B* satisfies the existential sentence φ_{12}^n the structure *A* satisfies φ_{12}^n too. It is easy to see that *A* reflects the whole computation of the machine *M* on input 1. If *M* halts at moment $T(n)$ then $q_m(T(n))$ holds in *A*. In virtue of φ_3 there is no element $u > T(n)$ in *A* that satisfies q_m. Since *A* satisfies φ^n there is some $x \leq T(n)$ in *A* that satisfies *Q*. Both $T(n)$ and *x* belong to *B*; hence *B* satisfies φ^n.

Note that φ_{12}^n does not depend on the choice of *g*. Define $g(n) = f(|\varphi_{12}^n| + n)$. Since φ_{12}^n is the only part of φ^n that depends on *n*, the number $k = |\varphi^n| - |\varphi_{12}^n|$ does not depend on *n*. Let $\varphi = \varphi^k$. Then $g(k) = f(|\varphi|)$. The computation of *M* on input 1^k halts at certain moment that will be denoted $T(k)$.

Finally let ψ be a universal sentence $\forall x_1 \ldots \forall x_l \ \psi'(x_l, \ldots, x_l)$ that is equivalent to φ in the finite case. Here ψ' is quantifier-free. Let *A* be the model of $\varphi_{01} \& \varphi_{02} \& \varphi_{11} \& \varphi_{12}^k \& \varphi_2 \& \varphi_3$ with the universe $(0, 1, \ldots, T(k))$ and the intended interpretation of the predicates. First we define *Q* to be empty in *A*. The resulting structure A_0 does not satisfy φ; hence it does not satisfy ψ and $\neg\psi'(c_1, \ldots, c_l)$ holds in A_0 for some $c_1, \ldots, c_l$. If $l < T(k)$ choose $c \in A - \{0, c_1, \ldots, c_l\}$ and put *c* into *Q*. The resulting structure A_1 satisfies φ yet $c_1, \ldots, c_l$ still witness failure of ψ in A_1 which is impossible. Thus $|\psi| \geq l \geq T(k) \geq g(k) \geq f(|\varphi|)$. □

§3. First-order logic versus second-order logic

In spite of the criticism in Sections 1 and 2, first-order logic is still *a* very good logic even in the case of finite structures. It is not without reason that first-order logic is used in computer science. It is elegant, natural and fairly expressive. However, if elegance, naturality and expressiveness are that important why wouldn't we turn to second-order logic? Second-order logic is elegant and natural as well, and it is much more expressive.

Second-order logic is not very popular among logicians. The objection against second-order logic is that it is not well manageable. However some fragments of second-order logic are much better manageable. One of them is weak second-order logic, which allows quantification over finite predicates only. In the finite case, of course, there is no difference between the two versions of second-order logic.

As we saw in §1 the theorems that made first-order logic so much preferable to second-order logic often fail or become meaningless in the finite case. Is there any important advantage of first-order logic versus second-order logic in the finite case? We take a computational point of view and answer this question positively.

Proviso 1. *The term "structure" refers to finite structures if the contrary has not been stated explicitly.*

A structure will be viewed as certain data, as an input to algorithms. A seeming difficulty is that elements of a structure are not necessarily constructive objects. We are interested however in the isomorphism type of a given structure rather than in the nature of its elements. Recall that $|S|$ is the cardinality of a structure *S*.

Proviso 2. *The universe of a structure* S *consists of numbers* $0, 1, \ldots, |S|-1$.

Proviso 2 by itself does not turn structures into inputs. We still have to choose a way to represent basic relations and functions. For example, a graph (V,E) may be represented as the lexicographically ordered list of edges or as an array $A(i, j)$ where $A(i, j) = 1$ if $(i, j) \in E$ and $A(i, j) = 0$ otherwise.

Proviso 3. *A reasonable standard way to represent structures is chosen.*

We introduce global predicates. Let σ be a vocabulary. An l-ary σ-predicate is a function π that assigns to each σ-structure S an l-ary g predicate π^S on S. (The superscript S will be usually omitted.) A zero-ary σ-predicate π assigns a truth value to each σ-structure and therefore can be viewed as the set {S: S is a σ-structure and π^S is true). Every first-order formula in the vocabulary σ with l free variables gives an l-ary σ-predicate. A global predicate is a σ-predicate for some σ.

Examples. Let $\sigma = \{E\}$ where E is a binary predicate symbol. Note that σ-structures are graphs. The first example is a binary σ-predicate π_1 such that for any graph G and any elements x, y of G, $\pi_1(x, y)$ holds in G iff there is an E-path from x to y. A more usual way to describe π_1 is just to say that π_1 is the binary σ-predicate "There is an E-path from x to y". The second example is the set π_2 of symmetric graphs. In other words, π_2 is a zero-ary σ-predicate such that π_2 holds in a graph G iff G is symmetric. Note that every relational query is a global predicate.

With each global predicate π we associate the problem of computing (or recognizing) π. It is a decision problem. An instance of this decision problem is a pair $(S, \bar{x})$ where S is a structure of the vocabulary of π and $\bar{x}$ is a tuple of elements of S whose length is the arity of π. The corresponding question is whether $\pi(\bar{x})$ holds in S. In order to avoid trivialities we suppose that the length of the presentation of S is at least $|S|$.

Theorem 1. *A Boolean combination of* PTIME *recognizable global predicates is a* PTIME *recognizable global predicate. If* $\pi(x_1, . . . , x_l , y)$ *is an* $(l + 1)$*-ary* PTIME *recognizable global predicate then* $\exists y\, \pi(x_1, . . . , x_l , y)$ *is an* l*-ary* PTIME *recognizable global predicate (with an obvious meaning). Every first-order global predicate is* PTIME *recognizable.*

Proof. The first statement is obvious. The compute the truth value of $\exists y \pi(\bar{x}, y)$ in S compute successively the truth values for $\pi(\bar{x}, 0)$, $\pi(\bar{x}, 1), . . . ,$ $\pi(\bar{x}, |S|-1)$ in S. Since atomic first-order predicates are PTIME computable (here we need reasonable standard representations of structures), the third statement follows from the first two. □

Some second-order global predicates are *NP*-complete. For example, the set of 3-colorable graphs – a well-known *NP*-complete set – is definable by a second-order sentence $\exists X \exists Y \exists Z \psi(X, Y, Z)$ where *X, Y, Z* are unary predicate variables and ψ is first-order. Attaching little gadgets to vertices it is easy to construct an *NP*-complete set of graphs definable by a second-order sentence $\exists X \psi(X)$ where *X* is a unary predicate variable and ψ is first-order. Thus there are second-order global predicates that are not PTIME recognizable unless P = NP.

It is almost a consensus in Theoretical Computer Science that PTIME computations are feasible whereas superpolynomial time computations are intractable, see [GJ], [HU]. In particular, Hopcoft and Ullman write the following. "Although one might quibble that an n^{57} step algorithm is not very efficient, in practice we find that problems in *P* usually have low-degree polynomial time solutions".

Thus first-order global predicates appear to be feasibly recognizable, whereas recognizing a second-order global predicate may be intractable. From our point of view, explicit PTIME recognizability is a decisive advantage of first-order logic versus second order logic.

Remark. Theorem 1 remains true if "PTIME" is replaced by "LOGSPACE". The same proof proves the new (and stronger) version of the theorem: just represent numbers in binary. Theorem 1 and the stronger version of it are well-known.

§4. Fixed points and polynomial time logic

Provisos 1-3 of §3 are in force.

As we saw above in §3, first-order global predicates are PTIME computable and even LOGSPACE computable. Unfortunately neither of these two statements can be reversed. For example, the property of graphs to be of even cardinality is recognizable by an obvious algorithm in linear time and logarithmic space. In virtue of the Lemma in §1 this property is not first-order.

A natural idea arises: to augment first-order logic by additional operators in order to express exactly the PTIME (LOGSPACE, etc.) computable global predicates. This is the idea reflected in the title: given a complexity level to tailor a logic expressing exactly the global predicates computable within the complexity level. Neil Immerman uses the word "capture" [Im2]. The problem is to capture a given complexity level by logical means. This section is devoted mainly to logic tailored for PTIME.

Remark. Actually it makes sense to generalize the notion of global predicate to the notion of global function and try to capture exactly the global functions computable within a given complexity level. Restricting attention to global predicates is even ridiculous if we see our logic as a notation system for algorithms or a potential programming language. Just imagine a programming language such that each program outputs only a boolean value. Global functions and functional (rather than predicate) logics are explored in [Gu3].

Let us start with a note that first-order expressible global predicates apparently do not form a natural complexity class. They certainly do not form a complexity class defined by Turing machines with bounds on time and/or space (see again the even cardinality example). A computational model which is much closer to first-order logic is that of uniform sequences of boolean circuits of constant depth, unbounded fan-in, and polynomial size. Modest extensions of first-order logic do capture natural circuit complexity classes, see [Im2] and especially [GL] in this connection.

If we consider *NP*, co-*NP* and higher levels of the polynomial hierarchy [St] as genuine complexity classes then second-order logic and some of its natural sub-logics do capture complexity classes. (When we speak about second-order logic we suppose that there are no third-order predicates or functions.) Recall that an existential second-order formula is a second-order formula $\varphi = \exists x_1, \dots, \exists x_k \psi$ where ψ is first-order and $x_1, \dots, x_k$ are predicate (or function) variables. The formula ψ may have free predicate and function variables as well as free individual variables.

Theorem 1. *A global predicate is computable in polynomial time by a nondeterministic Turing machine if and only if it is expressible by an existential second-order formula.*

Theorem 1 is due to Fagin [Fal] and is readily generalizable to capture co-*NP* and higher levels of the polynomial hierarchy. Actually Fagin did not seek to characterize *NP*. It was just the other way around. He sought to characterize existential second-order sentences (generalized spectra in his terminology). Theorem 1 grew from investigations on spectra ;of first-order sentences [Be, JS, Fal, Bo]. It looks pretty obvious today, and nondeterministic polynomial time computable global predicates are not necessarily feasible. However existential second-order logic does capture exactly the nondeterministic polynomial time computable global predicates and this fact inspired attempts to capture in a similar way deterministic PTIME computable global predicates. (About extending Fagin's result to richer logics and higher complexity classes see [St] and [CKS].)

Meantime Codd proposed the relational database model and used variations of first-order logic (relational algebra, relational calculus) as query languages [Ul]. The relational model was a big success. However, the first-order query languages were proven to be too restrictive in many applications. Attempts were made to enrich those languages by additional operators, most notably by the transitive closure operator [Zl] and the least fixed point operator [AU].

The transitive closure of a binary global predicate $\alpha(x, y)$ of some vocabulary σ is a global σ-predicate $\beta(x, y)$ such that for every σ-structure *S* the relation β^S is the transitive closure of the relation α^S. More generally one can speak about the transitive closure of a global predicate $\alpha(\bar{x}, \bar{y})$ where $\bar{x}, \bar{y}$ are tuple of individual variables of the same length [Im2]. In addition to $\bar{x}$ and $\bar{y}$, α may have individual parameters. First-order expressible global predicates are not closed under the transitive closure operator; see Appendix 2.

In a conversation with Andreas Blass the question of notation for the transitive closure of $\alpha(\bar{p}, \bar{x}, \bar{y})$ was raised. The naive notation $TC\alpha(\bar{p}, \bar{x}, \bar{y})$ is ambiguous. A possible unambiguous notation is

$$TC(\bar{x}, \bar{y}; \alpha(\bar{p}, \bar{x}, \bar{y}), u, v) \text{ or } TC_{\bar{x}, \bar{y}}(\alpha(\bar{p}, \bar{x}, \bar{y}), \bar{u}, \bar{v}) .$$

Here $\bar{x}$ and $\bar{y}$ are tuples of bound variables, $\bar{p}$ is a tuple of parameters, and $\bar{u}, \bar{v}$ are tuples of new free variables.

Let us define the least fixed point operator for global predicates. It will be convenient to view global predicates as global sets: a global l-ary predicate α of a vocabulary σ assigns a set $\alpha^S \subseteq S^l$ to each σ-structure *S*. We order global l-ary σ-predicates by inclusion: $\alpha \leq \beta$ if $\alpha^S \subseteq \beta^S$ for every σ-structure *S*. We say that a global σ-predicate α is *empty* if α is empty for every σ-structure *S*.

Definition. Let σ be a vocabulary, *P* be an additional predicate variable of some arity l, and $\pi(P)$ be a global l-ary predicate of the vocabulary $\sigma \cup \{P\}$. View $\pi(P)$ as an operator that, given a global l-ary σ-predicate α, produces a global l-ary σ-predicate $\pi(\alpha)$. A global l-ary σ-predicate α is a *fixed point* for $\pi(P)$ if $\alpha = \pi(\alpha)$, and α is the *least fixed point* for $\pi(P)$ if it is a fixed point and $\alpha \leq \beta$ for every fixed point β for $\pi(P)$.

Recall the notion of monotonicity of a first-order formula in a predicate variable defined in Appendix 1. This notion obviously generalizes to monotonicity of a global predicate in a predicate variable.

Claim 1. *Let* σ, P, $\boldsymbol{l}$ *and* $\pi(P)$ *be as in the definition above. Suppose that* $\pi(P)$ *is monotone in* P. *Then there is a unique least fixed point for* $\pi(P)$. *Moreover, let* $\alpha_0, \alpha_1, \alpha_2, \ldots$ *be global* l*-ary* σ*-predicates such that* α_0 *is empty and every* α_{m+1} *equals* $\pi(\alpha_m)$. *If* β *is the least fixed point for* $\pi(P)$ *and* S *is* σ*-structure then* $\beta^S = \alpha^S_m$ *where* $m=|S|^{\boldsymbol{l}}$. *Thus the least fixed point* *for* $\pi(P)$ *is* PTIME *computable if* $\pi(P)$ is.

The proof is clear. The claim appears in [AU] in terms of relational algebra. A transfinite induction generalizes the claim to infinite structures. In either form the claim is a special case of the classical theorem of Tarski [Tar].

Example 1 [AU]. The transitive closure of a global predicate $E(x, y)$ is the least fixed point with respect to P for

$$E(x, y) \vee \exists z\, [P(x, z) \,\&\, P(z, y)].$$

Example 2. The semigroup generated by a set A is the least fixed point with respect to P for

$$A(x) \vee \exists z \exists y\, [P(y) \,\&\, P(z) \,\&\, x = y \bullet z]$$

A possible notation for the least fixed point for a global l-ary predicate $\pi(P)$ with free individual variables $x_1, \ldots, x_l$ is

$$\mathrm{LFP}\,(P, x_1, \ldots, x_l\, ; \pi, y_1, \ldots, y_l).$$

It reflects the fact that LFP binds P and $x_1, \ldots, x_l$. The new individual variables $y_1, \ldots, y_l$ are free.

By the definition, LFP applies only to global predicates that are monotone in a given predicate variable. By Claim 1 in Appendix 1 the decision problem whether a given first-order formula is monotone in a given predicate variable, is unsolvable. This poses a difficulty in defining the extension of first-order logic by LFP. To overcome this difficulty Chandra and Harel [CH2] use positivity instead of mono-tonicity.

Let FO + LFP be the extension of first-order logic by the following formation rule. (For the sake of definiteness we assume that substitution of terms for free occurrences of individual variables is one of the first-order formation rules.)

LFP *formation rule*. Let P be a predicate variable of some arity l and let $\varphi(P, x_1, \ldots, x_l)$ be a well-formed formula. If all free occurrences of P in φ are positive and $y_1, \ldots, y_l$ are new individual variables then

$$\mathrm{LFP}\,(P, x_1, \ldots, x_l\, ;\; \varphi(P, x_1, \ldots, x_l), y_1, \ldots, y_l)$$

is a well-formed formula. All occurrences of *P* and $x_1, \ldots, x_l$ in the new formula are bound. If *Q* is a predicate variable different from *P* then every free (resp. bound) occurrence of *Q* in φ remains free (resp. bound), and every positive (resp. negative) occurrence of *Q* in φ remains positive (resp. negative). The only occurrences of individual variables $y_1, \ldots, y_l$ in the new formula are bound. (φ may have individual parameters. They remain free.) The meaning of the new formula is that the tuple $(y_1, \ldots, y_l)$ belongs to the least fixed point for $\varphi(P, x_1, \ldots, x_l)$.

Remark. Allowing individual parameters does not increase the expressive power of FO + LFP. For example, the formula

$$\mathrm{LFP}\ (P, y; E(u, y) \lor \exists z\ (P(z) \,\&\, E(z, y)), x).$$

is equivalent to the formula

$$\mathrm{LFP}(Q, w, y; E(w, y) \lor \exists z\ (Q(w, z) \,\&\, E(z, y)), u, x).$$

More generally,

$$\mathrm{LFP}(P, y; \varphi(P, u, y), x)$$

is equivalent to

$$\mathrm{LFP}(Q, w, y; \varphi(Q_w, w, y), u, x)$$

where $Q_w(z) = Q(w, z)$. However, parameters may be useful from the computational point of view.

Sometimes logicians speak about logic with equality. In those cases the equality relation is a logical constant. The equality sign is interpreted as the identity relation on the elements of a given structure and it is not listed as a member of a given vocabulary. By Proviso 2 our structures are built from natural numbers. This allows us to introduce the natural order of elements as a logical constant and to speak about logic with order.

Theorem 2 [Iml, Var]. *A global predicate is* PTIME *computable if and only if it is expressible in* FO + LFP *with order*.

The "if" implication of Theorem 2 follows from Theorem 1 in §3 and from Claim 1. A sketch of a proof of the "only if" implication can be found in [Iml]. An alternative proof of the "only if" implication will be indicated later in this section.

Aho and Ullman [AU] define a generalization of LFP whose application is not restricted by monotonicity. A similar idea was independently explored by Livchak [Li1]. Unaware of developments related to the least fixed point operator Livchak (who happened to be a former Ph.D. student of mine) proposes to augment the definition of first-order formulas by the following additional formation rule:

If $F(\bar{x})$, $G(\bar{x})$ and $H(\bar{x})$ are well-formed formulas with the same free individual variables $\bar{x} = (x_1, \ldots, x_l)$ then $L(F(\bar{x}), G(\bar{x}), H(\bar{x}))$ is a new well-formed formula whose meaning is the infinite disjunction

$$F_0(\bar{x}) \vee F_1(\bar{x}) \vee F_2(\bar{x}) \vee \ldots$$

where $F_0(\bar{x})$ is $H(\bar{x})$ and each $F_{i+1}(\bar{x})$ is the disjunction of $F_1(\bar{x})$ and the result of replacing each subformula $G(y_1, \ldots, y_l)$ of $F(\bar{x})$ by $F_i(y_1, \ldots, y_l)$.

The extension of first-order logic with order by Livchak's rule captures PTIME [Li2]. We incorporate this fact into Theorem 3. But first let us reformulate Livchak's rule.

Definition. Let σ be a vocabulary, P be an additional predicate variable of some arity l, and $\pi(P)$ be a global l-ary predicate of the vocabulary $\sigma \cup \{P\}$. View $\pi(P)$ as an operator that, given a global l-ary σ-predicate α, produces a global l-ary σ-predicate $\pi(\alpha)$. This operator $\pi(P)$ is *inflationary* if $\alpha \le \pi(\alpha)$ for every global σ-predicate α. Let α_0, α_1, etc. be a sequence of global l-ary σ-predicates where α_0 is empty and each α_{i+1} equals to $\pi(\alpha_i)$. A fixed point β for $\pi(P)$ is an *iterative fixed point* if for every σ-structure S there is an i with $\beta^S = \alpha^S_i$.

Claim 2. *Let* σ, P, $\boldsymbol{l}$, $\pi(P)$ *and* α_0, α_1, *etc. be as in the definition above. Suppose that* $\pi(P)$ *is inflationary in* P. *Then there is a unique iterative fixed point* β *for* $\pi(P)$. *Moreover, for every* σ-*structure* S, $\beta^S = \alpha^S_m$ *where* $m=|S|^{\boldsymbol{l}}$.

Thus the iterative fixed point for $\pi(P)$ *is* PTIME *computable if* $\pi(P)$ is.

The proof is clear. Note that if P and $\pi(P)$are as in the definition above then $P \vee \pi(P)$ is inflationary. Let FO + IFP be the extension of first-order logic by the following formation rule.

IFP *formation rule*. Let P be a predicate variable of some arity l, and let $\varphi(P, \bar{x})$ be a well-formed formula whose free individual variables are all or some members of $\bar{x} = (x_1, \dots, x_l)$. If $\bar{y} = (y_1, \dots, y_l)$ is a tuple of new individual variables then

$$\mathrm{IFP}(P, \bar{x}; P(\bar{x}) \vee \varphi(P, \bar{x}), \bar{y})$$

is a well-formed formula. The meaning of the new formula is that $\bar{y}$ is in the iterative fixed point for $P(\bar{x}) \vee \varphi(P, \bar{x})$.

Claim 3. FO + IFF *expresses exactly the global predicates expressible in first-order logic augmented by Livchak' s rule.*

Proof. We consider the extension of first-order logic by both formation rules and show that either rule can be eliminated. The formula

$$\mathrm{IFP}(P, \bar{x}; P(\bar{x}) \vee \varphi(P, \bar{x}), \bar{y})$$

is equivalent to

$$L(\varphi(P, \bar{y}), P(\bar{y}), \mathrm{FALSE}(\bar{y})).$$

Given a formula $L(F, G, H)$ with free individual variables $\bar{x} = (x_1, \dots, x_l)$ and an additional l-ary predicate variable P write down a formula $F'(P, \bar{x})$ such that $F(\bar{x}) = F'(G, \bar{x})$. Using P, $F'(P, \bar{x})$, $H(\bar{x})$ and first-order means write down a formula $\varphi(P, \bar{x})$ saying the following:

If $\neg\exists\bar{x}\, H(\bar{x})$ then $F'(P, \bar{x})$,

else if $\neg\exists\bar{x}\, P(\bar{x})$ then $H(\bar{x})$,

else $F'(P, \bar{x})$.

It is easy to check that $L(F(\bar{x}), G(\bar{x}, H(\bar{x}))$ is equivalent to

$$\mathrm{IFP}(P, \bar{y}; P(\bar{y}) \vee \varphi(P, \bar{y}), \bar{x})$$

where $\bar{y}$ is a tuple of new individual variables. □

Theorem 3. *Let it be a global predicate. The following statements are equivalent*:

(1) π *is* PTIME *computable*,

(2) π *is expressible in* FO + LFP *with order, and*

(3) π *is expressible in* FO + IFF *with order*.

Proof. The implication (1) $\rightarrow$ (2) follows from Theorem 2. The implication (3) $\rightarrow$ (1) follows from Claim 2. To prove the implication (2) $\rightarrow$ (3) note that if a global *l*-ary predicate $\pi(P, \bar{x})$ is monotone in an *l*-ary predicate variable *P* then

$$\mathrm{LFP}(P, \bar{x}; \pi(P, \bar{x}), \bar{y})$$

is equivalent to

$$\mathrm{IFP}(P, \bar{x}; P(\bar{x}) \vee \pi(P, \bar{x}), \bar{y}). \qquad \square$$

Chandra and Harel show that FO + LFP without order is not able to express the global zero-ary predicate "The cardinality of a given structure is even" [CH2]. Their argument can be extended to show that FO + IFF without order is not able to express the same global predicate. Our Appendix 3 gives an alternative proof of the fact that FO + IFP without order is not able to express some PTIME computable order-independent global predicates π.

We turn our attention to global functions.

Definition. *A* global partial function *f* of vocabulary σ, arity *l* and co-arity *r* assigns to each σ-structure *S* a partial function f^S from S^l to S^r.

Example 3. Let σ consist of one binary predicate variable *E* (for "edge"), so that σ-structures are (directed) graphs. Let f(x, y) be the length of a shortest path from *x* to *y*. If *S* is a graph and f^S is defined at (x, y) then $f^S(x, y) < |S|$ and therefore $f^S(x, y)$ is an element of *S*.

Example 4. Let again σ be the vocabulary of graphs. For every graph *S* and every $x \in S$ let f(x) be the pair $(y, z) \in S^2$ such that there are exactly $y \cdot |S| + z$ elements *u* with an edge from *x* to *u*.

As was mentioned above, we are interested in logics (or algebras) that capture PTIME (LOGSPACE, etc.) computable global functions. In a sense FO + LFP with order does capture PTIME computable functions: it allows one to speak about the graph of a PTIME computable function f and about digits in the binary notation for $f(\bar{x})$. We prefer to speak about global functions directly. See [Gu3] in this connection. Here we mention only the results related to PTIME.

Let us ignore singleton structures (alternatively we may allow boolean variables). See the definition of recursive global partial functions in [Gu3].

Theorem 4 [Gu3, Sa]. *A global partial function is* PTIME *computable if and only if it is recursive*.

Two algebras of recursive global partial functions were given in [Gu3] by some initial members and certain operations. Let ARF (for "Algebra of Recursive Functions") be either of them.

Theorem 5 [Gu3]. *A global partial function is* PTIME *computable if and only if it belongs to* ARF.

An important advantage of (the proof of) Theorem 5 versus (the proof of) Theorem 2 is preserving essential time bounds.

Remark. It is easy to prove directly that the graph of every function in ARF is expressible in FO + LFP with order. This together with Theorem 5 gives an alternative proof of the "only if" implication of Theorem 2. Let $\pi(x)$ be a PTIME computable global predicate and let $f(\overline{x})$ be the characteristic function for $\pi(x)$ i.e. $f(\overline{x}) = 1$ if $\pi(x)$ holds and $f(\overline{x}) = 0$ otherwise. By Theorem 5, f is in ARF. Hence the predicate $f(\overline{x}) = y$ is expressible in FO + LFP with order. Hence the predicate $f(\overline{x}) = 1$ is expressible in FO + LFP with order.

§5. Interpolation and definability for polynomial time logic

According to §1, many famous and important theorems about first-order logic fail in the case of finite structures. What happens to those theorems in the case of logic tailored for polynomial time? We concentrate here on the interpolation and definability principles for polynomial time logic and show that these principles are equivalent to natural complexity principles whose status is unknown.

Let PTL (for Polynomial Time Logic) be the logic FO + LFP with order, or the logic FO + IFP with order, or an algebra of PTIME computable functions from [Gu3]. It will be important that PTL expresses precisely PTIME computable global predicates. The exact syntax of PTL will not be important.

Definition. A partial function f from $\{0, 1\}^*$ to $\{0, 1\}^*$ is *polynomially bounded* if there is a natural number k such that $|f(x)| \leq |x|^k$ for all $x \in \mathrm{Domain}(f)$ with $|x| > 1$. Here $|x|$ is the length of x. More generally, a binary relation B over $\{0, 1\}^*$ is *polynomially bounded* if there is k such that $B(x, y)$ and $|x| > 1$ imply $|y| \leq |x|^k$.

We identify a nonempty word $x = a_0 a_1 \ldots a_{l-1}$, in $\{0, 1\}^*$ with the structure with universe $\{0, 1, \ldots, \boldsymbol{l}-1\}$ and one unary predicate $X = \{i: a_i = 1\}$. If $\boldsymbol{l} > 2$, $m = \boldsymbol{l}$ for some k and y is a word $b_0, b_1, \ldots, b_{\mathbf{m-1}}$ in $\{0, 1\}^*$, we identify the pair (x, y) with the extension of the structure x by a k-ary predicate

$Y = \{(i_1, \ldots, i_k)$: if j is the number whose notation in the positional system of base $\boldsymbol{l}$ is $i_1 \ldots i_k$ then $b_j = 1.\}$

Lemma 1. *For every* NP *set* A *of nonempty words over* $\{0, 1\}^*$ *there is a* PTL *sentence* φ *such that*

$A = \{x \in \{0, 1\}^*$: x is the reduct of a model of $\varphi\}$.

Proof. Without loss of generality, every $x \in A$ is of length at least 2. There are a natural number k and a PTIME computable polynomially bounded binary relation B over $\{0, 1\}^*$ such that $A = \{x: (x, y) \in B$ for some $y\}$, and $(x, y) \in B$ implies $|y| = |x|^k$. The desired sentence φ expresses $(x, y) \in B$. □

The analogue of Craig's Interpolation Theorem for PTL will be called the *Interpolation Principle* for PTL. This principle states that for every valid (in all relevant finite structures) PTL sentence $\varphi_1 \rightarrow \varphi_2$ there is a PTL sentence θ such that

$$\text{vocabulary}(\theta) \subseteq \text{vocabulary}(\varphi_1) \cap \text{vocabulary}(\varphi_2)$$

and the implications $\varphi_1 \rightarrow \theta$, $\theta \rightarrow \varphi_2$ are valid.

Theorem 1 . *The following two statements are equivalent:*

(1) *The Interpolation Principle for* PTL, *and*

(2) *The following separation principle for* NP: *for every pair of disjoint* NP *subsets* A_1, A_2 *of* $\{0, l\}^*$ *there is a* P *subset* B *of* $\{0, l\}^*$ *such that* B *includes* A_1 *and avoids* A_2.

Proof. First suppose (1) and let A_1, A_2 be disjoint NP subsets of $\{0, 1\}^*$. Without loss of generality neither A_1 nor A_2 contains the empty word. By Lemma 1 there are PTL sentences φ_1, φ_2 such that $A_i = \{x \in \{0, l\}^*$: x is the reduct of a model of $\varphi_i\}$ for $I = l, 2$. Without loss of generality, the only common non-logical constant of φ_1, φ_2 is the unary predicate variable X. Obviously, the implication $\varphi_1 \rightarrow \neg\varphi_2$ is valid. Let θ be an appropriate interpolant. The set of models of θ is the desired set B.

Next suppose (2) and let $\varphi_1 \rightarrow \varphi_2$ be a valid PTL sentence. Let σ be the common part of the vocabularies of φ_1, φ_2. For i =l, 2 let

$A_i = \{x$: x is the binary code for the σ-reduct of a model of $\varphi_i\}$.

By (2) there is a P set B that includes A_1 and avoids A_2. The desired interpolant θ expresses $x \in B$. □

Note that the Interpolation Principle for PTL implies NP $\cap$ co-NP = P.

The analogue of Beth Definability Theorem for PTL will be called the *Definability Principle* for PTL. This principle states the following. Let σ be a vocabulary, P be an additional predicate variable of some arity $\boldsymbol{l}$ and $\varphi(P)$ be a PTL sentence of the vocabulary $\sigma \cup \{P\}$. Suppose that for every σ-structure S and all $P_1, P_2 \subseteq S^{\boldsymbol{l}}$, $\varphi^S(P_1)$ and $\varphi^S(P_2)$ imply $P_1 = P_2$. Then there is a PTL formula ψ of the vocabulary σ with $\boldsymbol{l}$ free variables such that for every

σ-structures S and every $P_1 \subseteq S^l$,

$$\varphi^S(P_1) \text{ implies } P_1 = \psi^S.$$

The *Weak Definability Principle* for PTL is the result of strengthening the antecedent of the Definability Principle for PTL as follows: for every σ-structure S there is a unique $P_1 \subseteq S^l$ such that $\varphi^S(P_1)$ holds.

Definition (cf. [Val]). A nondeterministic Turing machine M is *unambiguous* if for every input x there is at most one accepting computation of M on x. An NP subset A of $\{0, 1\}^*$ is UNAMBIGUOUS if there is an unambiguous Turing machine that accepts A.

Theorem 2. *The following statements* (l) - (4) *are equivalent.*

(1) *The Definability Principle for* PTL.

(2) *For every polynomially bounded partial function* f *from* $\{0, 1\}^*$ *to* $\{0, 1\}^*$, *if the graph of* f *is in* P *then* f *is* PTIME *computable.*

(3) *For every polynomially bounded partial function* f *from* $\{0, 1\}^*$ *to* $\{0, 1\}^*$, *if th e graph of* f *is in* P *then the domain of* f *is in* P.

(4) UNAMBIGUOUS = P.

Proof. (1) $\rightarrow$ (2) . Suppose (1) and let f be a polynomially bounded partial function from $\{0, 1\}^*$ to $\{0, 1\}^*$ with PTIME computable graph. It suffices to construct a PTIME algorithm for calculating f(x) for x of length at least 2. Let x range over words of length at least 2. Without loss of generality there is k such that $|f(x)| = |x|^k$ for all x in Domain(f). There is a PTL sentence $\varphi(X, Y)$ with a unary predicate variable X and a k-ary predicate variable Y that expresses $(x, y) \in \text{Graph}(f)$. By (1) there is a PTL formula ψ such that if $\varphi(X, Y)$ holds in the structure (x, y) then

$$Y = \{(i_1, \ldots, i_k) \colon \psi(i_1, \ldots, i_k) \text{ holds in } x\}.$$

Here is a PTIME algorithm for calculating f(x). View x as a structure in the vocabulary $\{X\}$. Compute

$$Y = \{(i_1, \ldots, i_k) \colon \psi(i_1, \ldots, i_k) \text{ holds in } x\}.$$

The extension of the structure *x* by the predicate *Y* corresponds to a pair (x, y) for some word y of length $|x|^k$. Check whether $\varphi(X, Y)$ holds in the extended structure. If yes then y = f (x) .

The implications (2) → (3) and (3) → (4) are trivial.

(4) → (1). Suppose (4) and let σ be a vocabulary

variable of some arity *i* and $\varphi(P)$ be a PTL sentence such that for every σ-structure *S* there is at most one PCS satisfying $\varphi^S(P)$. Set

$$K = \{(S, \bar{c}): S \text{ is a } \sigma\text{-structure, } c \in S^{l} \text{ and there is } P \subseteq S^{l} \text{ such that } \varphi^S(P;) \text{ holds and } \bar{c} \in P\}.$$

Obviously, *K* is UNAMBIGUOUS. By (4), *K* is *P*. The desired PTL formula $\psi(v_1, \ldots, v_l)$ expresses $(S, v_1, \ldots, v_l) \in K$. □

The following theorem was established in a discussion with Neil Immerman. (It succeeded Theorem 1 but preceded Theorem 2.)

Theorem 3. *The following statements* (1) - (3) *are equivalent.*

(1) *The Weak Definability Principle for* PTL.

(2) *For every polynomially bounded function* $f:\{0, l\}^* \to \{0, 1\}^*$, *if the graph of* f *is in* P *then* f *is* PTIME *computable.*

(3) UNAMBIGUOUS ∩ co-UNAMBIGUOUS = P.

Proof. The case (1) → (2) is similar to the case (1) → (2) in the proof of Theorem (2) .

(2) → (3). Suppose (2) and let A_0, A_1 be complementary UNAMBIGUOUS subsets of {0, 1}*. There are unambiguous nondeterministic Turing machines M_0, M_1 accepting A_0, A_1 respectively. For i = 0, l and $x \in A_i$ let f(x) be the digit *I* followed be the binary code for the accepting computation of M_1 on *x*. By (2), *f* is PTIME computable. Hence A_0 and A_1 are *P*.

(3) → (1). Suppose (3) and let σ, *P*, $\varphi(P)$ and *K* be as in the case (4) → (1) of the proof of Theorem 2 except now for every σ-structure *S* there is a unique $P \subseteq S^{l}$ satisfying $\varphi^S(P)$. Obviously *K* is UNAMBIGUOUS and co-UNAMBIGUOUS. By (3), *K* is *P*. The desired PTL formula $\psi(v_1, \ldots, v_l)$ expresses $(S, v_1, \ldots, v_l) \in K$. □

As we saw in §1, the Interpolation Principle for first-order logic implies the Definability Principle for first-order logic. The same proof shows that the Interpolation Principle for PTL implies the Definability Principle for PTL. If P = NP then the Interpolation Principle for PTL is obviously true. It is easy however to construct an oracle for which even the Weak Decidability Principle for PTL fails.

Claim (Andreas Blass) . *There is an oracle for which the Weak Definability Principle for* PTL *fails.*

Proof. By Theorem 3 it suffices to construct an oracle A and a function f from $\{0, 1\}^*$ (or from a P subset of $\{0, 1\}^*$) to $\{0, 1\}^*$ such that f is not PTIME computable relative to A whereas the graph of f is. We construct $A \subseteq \{0, l\}^*$, containing exactly one word w_n of each length n, such that the function $f(0^n) = w_n$ is not PTIME computable relative to A.

Enumerate all (deterministic) PTIME bounded query machines. Let P_k, be the time bound for a machine M_k. We define A in stages, choosing finitely many w_n' s at each stage. The kth stage will ensure that M_k^A does not compute f.

Stage k. Fix a natural number d that is larger than any n for which w_n has already been chosen and so large that $p_k(d) \leq 2^d - 2$. Set $w_n = 0$ for all $n < d$ for which w_n was not previously chosen. Run M_k with input 0^d and oracle $\{w_n : n < d\}$. Let B be the set of queries during the computation. By the time bound, $|B| \leq p_k(d) \leq 2^d - 2$. Choose w_d in $\{0, 1\}^d - B$ such that w_d differs from the output (if any) of M_k. If $d < l$ and $l < |x|$ for some $x \in B$ chose w_l in $\{0, 1\}^l - B$. It is easy to see that M_k^A will not compute w_d, on input 0^d. □

The computational status of Craig's Interpolation Theorem for prepositional logic was explored by Mundici [Mu] .

Appendix 1. Monotone versus positive.

This appendix is devoted to an important theorem about first-order logic whose status in the finite case is unknown.

Definition. Let σ be a vocabulary, P be an additional predicate variable of arity l and $\varphi(P, x_1, \ldots, x_r)$ be a first-order formula in the vocabulary $\sigma \cup \{P\}$ with free individual variables as shown. The formula φ is *monotonically increasing* in P if every σ-structure S satisfies the following for every l-ary predicates P_1, P_2 on S.

$$\text{if } \forall x_1 \ldots \forall x_l \, [P_1(x_1, \ldots, x_l) \rightarrow P_2(x_1, \ldots, x_l)]$$

$$\text{then } \forall x_1 \ldots \forall x_r \, [\varphi(P_1, x_1, \ldots, x_r) \rightarrow \varphi(P_2, x_1, \ldots, x_r)].$$

Define in the obvious way the following: φ is *monotonically decreasing in* P, φ is *monotonically increasing in* P *on finite structures* (or, *in the finite case*), φ is *monotonically decreasing in* P on *finite structures.* We restrict our attention to monotonically increasing behavior; the generalization for monotonically decreasing behavior will be obvious. We say "monotone" for "monotonically increasing".

We say that a first-order formula φ is *positive in* a predicate symbol P if every appearance of P in φ is positive. A precise definition of positive appearances of a predicate symbol in a first-order formula can be found in [CK]. It is easy to see that φ is monotone in P if φ is positive in P.

Theorem 1. *If a first-order formula* φ *is monotone is a predicate symbol* P *then there is a first-order formula* φ' *such that* φ' *is equivalent to* φ *and positive in* P.

I do not know who was the first to formulate this theorem but it is an obvious consequence of the Lyndon Interpolation Theorem [CK].

Conjecture. *Theorem* 1 *fails in the case of finite structures.*

The rest of this appendix contains a few remarks related to the conjecture. First we exhibit a sentence which is monotone in a unary predicate symbol P on finite structures but which is not monotone in P in general. Let f be a unary function symbol. The desired sentence (with equality) says that f is one-to-one and that P is closed under f-predecessors if P is closed under f-successors.

Claim 1. *Let* P *be a predicate variable. The following problems are undecideable:*

(i) *Given a first-order sentence* φ *tell whether* φ *is monotone in* P, *and*

(ii) *Given a first-order sentence* φ *tell whether* φ *is monotone in* P *on finite structures.*

Proof. Without loss of generality P is just a propositional variable. Let ψ be a first-order sentence that does not mention P. It is valid (resp. valid in the finite case) iff the sentence $P \rightarrow \psi$ is monotone (resp. monotone on finite structures) in P. □

Corollary 1. *Let* P *be a predicate variable. There is no recursive function* f *from first-order sentences to first-order sentences such that an arbitrary first-order sentence* φ *is monotone in* P *if and only if the sentence* f(φ) *is positive in* P.

Corollary 2. *Let* P *be a predicate variable. There is no partial recursive function* f *from first-order sentences to first-order sentences such that an arbitrary first-order sentence* φ *is monotone in* P *on finite structures if and only if* f(φ) *is positive in* P.

In the case of a propositional variable P there is a simple function that, given a first-order sentence $\varphi(P)$, produces a first-order sentence $\varphi'(P)$ such that $\varphi'(P)$ is positive in P and $\varphi'(P)$ is logically equivalent to $\varphi(P)$ if $\varphi(P)$ is monotone in P. The desired $\varphi'(P)$ is $\varphi(\text{False}) \vee [P \,\&\, \varphi(\text{True})]$.

On the other hand, a routine coding, given an arbitrary first-order formula with a predicate variable P, produces a first-order formula φ' in the vocabulary $\{E,Q\}$ such that E is a binary predicate symbol, Q is a unary predicate symbol and φ is monotone (respectively, positive) in P iff φ' is monotone (respectively, positive in Q. Moreover, it can be ensured that φ' has a conjunct saying that E is symmetric and irreflexive.

I have also checked that Theorem 1 remains true in the case of finite structures if φ is an existential sentence, a universal sentence, a prenex sentence with prefix $\exists^n\forall$ or a prenex sentence with prefix $\forall^n\exists$. Andreas Blass observed that if $\varphi(P)$ is positive (resp. monotone, monotone on finite structures) in P then so is $\neg\varphi(\neg P)$. Thus, if Theorem 1 is true in the finite case for prenex sentences φ with certain prefixes then it is true in the finite case for prenex sentences with the dual prefixes.

Appendix 2. Transitive closure is not first-order expressible.

Theorem. *Connectivity of a given graph is not first-order expressible in the case of finite structures (even in the presence of 1inear order). Hence the transitive closure of a given binary relation is not first-order expressible in the case of finite structures (even in the presence of linear order).*

Proof. For every positive integer n let S_n be the set $\{0,1, \dots ,n\text{-}l\}$ with the natural order and $E_n(x, y)$ be the following binary relation on $\{0,1, \dots ,n\text{-}l\}$: either $y = x+2$ or $x = n\text{-}l$ and $y = 0$. Note that a graph (S_n, E_n) is connected iff n is even, and a relation E_n is obviously and uniformly in n expressible in the first-order language of order. Now use Lemma(ii) in §1. □

The theorem (without mentioning linear order) is due to Fagin [Fa2] and was reproved several times. Gaifman and Vardi wrote even a special paper [GV] with a specially short proof and a brief review of other known proofs. Fagin's proof actually gives more: connectivity is not expressible by an existential second-order sentence where all quantified predicate variables are unary. Yiannis Moschovakis noticed that all those proofs do not work in the presence of linear order and expressed an interest in such results in the presence of linear order.

Appendix 3. The fixed point operators can be powerless.

Provisos 1-3 of §3 are in force here because we will consider structures as inputs for algorithms. On the other hand the natural order of elements of a given structure is not a logical constant here, so that isomorphisms can break the order of elements. We are interested more in isomorphism types of structures than in specific representations.

Definition. A global l-ary predicate it of some vocabulary a is *invariant* if for every isomorphism f from a σ-structure A onto a σ-structure B and every l-tuple $\bar{a} \in A$, $\pi^A(\bar{a})$ is equivalent to it $\pi^B(f\bar{a})$.

Any first-order expressible global predicate is invariant as well as any global predicate expressible in FO + LFP or FO + IFP. All these global predicates are PTIME computable. They do not exhaust, however, the PTIME computable global predicates. The following theorem provides plenty of counterexamples. It speaks about FO + IFP because FO + IFP subsumes FO + LFP. Recall that a first-order theory T is called ω -categorical if all countable models of T are isomorphic.

Theorem 1. *Let T be an* ω-*categorical first-order theory of some vocabulary* σ. *Then for every formula* $\varphi(\bar{x})$ *in* FO + IFP *there is a first-order formula* $\varphi'(\bar{x})$ *such that the global predicates* $\varphi(\bar{x})$ *and* $\varphi'(\bar{x})$ *coincide on the finite models of T.*

Remark. Actually, the global predicates $\varphi(\bar{x})$ and $\varphi'(\bar{x})$ will coincide on all models of T but we do not care here about infinite models.

Proof. Without loss of generality the given formula $\varphi'(\bar{x})$ is

$$\text{IFP}(P, \bar{y}; P(\bar{y}) \lor \psi(P, \bar{y}), \bar{x})$$

where ψ is first-order. Let $\alpha_0(\bar{x})$ = FALSE and let each $\alpha_{i+1}(\bar{x}) = \alpha_i(\bar{x}) \lor \psi(\alpha_i, \bar{x})$. By Ryll-Nardzewski's Theorem, there is a finite p such that $\alpha_{p+1}(\bar{x})$ is equivalent to $\alpha_p(\bar{x})$ on the infinite models of T. By the compactness theorem, there is a finite m such that $\alpha_{p+1}(\bar{x})$ is equivalent to $\alpha_p(\bar{x})$ on all models of T of size at least *m*. Hence there is a finite *q* such that $\alpha_{q+1}(\overline{x})$ is equivalent to $\alpha_q(\overline{x})$ in *T*. The first-order formula α_q is the desired ψ. □

Now we are ready to show that some polynomial time computable invariant predicates are not expressible in FO + IFP.

Example 1. The first-order theory of equality is ω-categorical. The predicate

$$\pi(A) = \begin{cases} \mathbf{0} & \textbf{if } |A| \textbf{ even} \\ \mathbf{1} & \textbf{otherwise} \end{cases}$$

is not first-order (see Lemma in §1). By Theorem 1, it is not expressible in FO + IFP. (Cf. [CH2, Theorem 6.2].)

Example 2. Let *F* be a finite field. Let *T* be the first-order theory of vector spaces over *F*. Obviously, *T* is ω-categorical. Let π be the global predicate such that for every structure *A* of the vocabulary of *T*, $\pi^A = 1$ if *A* is a model of *T* and dimension(*A*) is even, and $\pi^A = 0$ otherwise. Using Ehrenfeucht games [Eh] it is easy to check that π is not first-order. By Theorem 1 it is not expressible in FO + IFP.

It is not difficult to extend FO + LFP in such a way that the predicates of Examples 1 and 2 are expressible in the extended logic and only polynomial time computable invariant predicates are expressible in the extended logic. Still, the problem to design a logic that expresses exactly polynomial time computable invariant global predicates is open.

References

[Au] A.V. Aho and J.D. Ullman, "Universality of Data Retrieval Languages", Proc. of 6th ACM Symposium on Principles of Programming Languages, 1979, 110-117.

[Be] J. Bennet, "On Spectra", Doctoral Dissertation, Princeton University, N.J. 1962.

[Bo] E. Börger, "Spektralproblem and Completeness of Logical Decision Problems", in "Logic and Machines: Decision Problems and Complexity (ed. E. Börger, G. Hasenjaeger, D. Rödding), Springer Lecture Notes in Computer Science, to appear.

[CH1] A.K. Chandra and D. Harel, "Computable Queries for Relational Databases", Journal of Computer and System Sciences 21 (1980), 156-178.

[CH2] A.K. Chandra and D. Harel, "Structure and Complexity of Relational Queries", Journal of Computer and System Sciences 25 (1982), 99-128.

[CK] C.C. Chang and H.J. Keisler, "Model Theory", North-Holland, 1977.

[CKS] A.K. Chandra, D.C. Kozen and L.J. Stockmeyer, "Alternation", Journal of the ACM 28 (1981), 114-133.

[Co] K. Compton, A private communication.

[Cr] W. Craig, "Three Uses of the Herbrand-Gentzen Theorem in Relating Model Theory and Proof Theory", J. Symbolic Logic 22 (1957), 250-268.

[Eh] A. Ehrenfeucht, "An Application of Games to the Completeness Problem for Formalized Theories", Fund. Math. 49 (1961), 129-141.

[Fal] R. Fagin, "Generalized First-Order Spectra and Polynomial-Time Recognizable Sets", in "Complexity of Computation" (ed. R. Karp), SIAM-AMS Proc. 7 (1974), 43-73.

[Fa2] R. Fagin, "Monadic generalized spectra", Zeitschr. f. math. Logik und Grundlagen d. Math. 21, 1975, 89-96.

[Fr] H. Friedman, "The Complexity of Explicit Definitions", Advances in Mathematics 20 (1976), 18-29.

[FV] S. Feferman and R.L. Vaught, "The First-Order Properties of Algebraic Systems", Fund. Math. 47 (1959), 57-103.

[GJ] M.R. Garey and D.S. Johnson, "Computers and Intractability: A Guide to the Theory of NP-completeness", Freeman, 1979.

[GKLT] Yu. V. Glebskii, D.I. Kogan, M.I. Liogon'kii and V.A. Talanov, "Range and Degree of Relizability of Formulas in the Restricted Predicate Calculus", Cybernetics 5:2 (1969), 17-28; translation (1972), 142-154.

[GL] Y. Gurevich and H.R. Lewis, "A Logic for Constant-Depth Circuits", Manuscript, 1983.

[GV] H. Gaifman and M.Y. Vardi, "A Simple Proof that Connectivity is not First-Order", Manuscript, 1983.

[Go] W. Goldfarb, "The Gödel Class with Identity is Unsolvable", Journal of Symbolic Logic, to appear.

[Gul] Y. Gurevich, "Existential Interpretation", Algebra and Logic, 4:4 (1965), 71-85 (or "Existential Interpretation II", Archiv Math. Logik 22 (1982), 103-120).

[Gu2] Y. Gurevich, "Recognizing Satisfiability of Predicate Formulas", Algebra and Logic 5:2 (1966), 25-55; "The Decision Problem for Logic of Predicates and Operations", Algebra and Logic 8 (1969), pages 160-174 of English translation; "The Decision Problem for Standard Classes", Journal of Symbolic Logic 41 (1976), 460-464.

[Gu3] Y. Gurevich, "Algebras of Feasible Functions", Proc. of 24th IEEE Symposium on Foundations of Computer Science, 1983, 210-214.

[HU] J.E. Hopcroft and J.D. Ullman, "Introduction to Automata Theory, Languages and Computation", Addison-Wesley, 1979.

[Iml] N. Immerman, "Relational Queries Computable in Polynomial Time", Proc. of 14th ACM Symposium on Theory of Computing, 1982, 147-152.

[Im2] N. Immerman, "Languages which Capture Complexity Classes", Proc. of 15th ACM Symposium on Theory of Computing, 1983, 347-354.

[JS] N.G. Jones and A.L. Selman, "Turing Machines and the Spectra of First-Order Formulas", Journal of Symbolic Logic 39 (1974), 139-150.

[Ko] V.F. Kostyrko, "To the Decision Problem for Predicate Logic", Ph.D. Thesis, Kiev, 1965 (Russian).

[Kr] G. Kreisel, Technical Report No. 3, Applied Mathematics and Statistics Labs., Stanford University, January 1961.

[Lil] A.B. Livchak, "The Relational Model for Systems of Automatic Testing", Automatic Documentation and Mathematical Linguistics 4 (1982) , pages 17-19 of Russian original.

[Li2] A.B. Livchak, "The Relational Model for Process Control", Automatic Documentation and Mathematical Linguistics 4 (1983), pages 27-29 of Russian original.

[Lin] P. Lindström, "On Extensions of Elementary Logic", Theoria 35 (1969), 1-11.

[Mu] D. Mundici, "Complexity of Craig's Interpolation", Annales Societ. Math. Polonae, Series IV: Fundamenta Informaticae, vol. 3-4 (1982); "*NP* and Craig's Interpolation Theorem", Proc. of Florence Logic Colloquium 1982, North Holland, to appear.

[Sa] V. Sazonov, "Polynomial Computability and Recursivity in Finite Domains", Elektronische Informationsverarbeitung and Kybernetik 16 (1980), 319-323.

[Sl] L.J. Stockmeyer, "The Polynomial - Time Hierarchy", Theoretical Computer Science 3 (1977), 1-22.

[Tai] W.W. Tait, "A Counterexample to a Conjecture of Scott and Suppes", Journal o£ Symbolic Logic 24 (1959), 15-16.

[Tar] A. Tarski, "A Lattice-Theoretical Fixpoint Theorem and its Application", Pacific Journal of Mathematics 5 (1955), 285-309.

[Tr] B.A. Trakhtenbrot, "Impossibility of an Algorithm for the Decision Problem on Finite Classes", Doklady 70 (1950), 569-572.

[Ul] J.D. Ullman, "Principles of Database Systems", Computer Science Press, 1982.

[Val] L.G. Valiant, "Relative Complexity of Checking and Evaluating", Information Processing 5 (1976), 20-23.

[Va] M.Y. Vardi, "Complexity of Relational Query Languages", Proc. of 14th ACM Symposium on Theory of Computing, 1982, 137-146.

[Zl] M.M. Zloof, "Query-by-Example: a Database Language", IBM Syst. Journal 16 (1977), 324-343.