INFORMATION AND CONTROL 67, 70–90 (1985)

A Zero–One Law for Logic with a Fixed-Point Operator

ANDREAS BLASS

Department of Mathematics,
University of Michigan, Ann Arbor, Michigan 48109

YURI GUREVICH

Department of Electrical Engineering and Computer Science,
University of Michigan, Ann Arbor, Michigan 48109

AND

DEXTER KOZEN

IBM Research, Yorktown Heights, New York 10598

The logic obtained by adding the least-fixed-point operator to first-order logic was proposed as a query language by Aho and Ullman (*in* "Proc. 6th ACM Sympos. on Principles of Programming Languages," 1979, pp. 110–120) and has been studied, particularly in connection with finite models, by numerous authors. We extend to this logic, and to the logic containing the more powerful iterative-fixed-point operator, the zero-one law proved for first-order logic in (Glebskii, Kogan, Liogonki, and Talanov (1969), *Kibernetika* **2**, 31–42; Fagin (1976), *J. Symbolic Logic* **41**, 50–58). For any sentence φ of the extended logic, the proportion of models of φ among all structures with universe $\{1, 2, ..., n\}$ approaches 0 or 1 as n tends to infinity. We also show that the problem of deciding, for any φ, whether this proportion approaches 1 is complete for exponential time, if we consider only φ's with a fixed finite vocabulary (or vocabularies of bounded arity) and complete for double-exponential time if φ is unrestricted. In addition, we establish some related results.

INTRODUCTION

Many statements about finite structures satisfy the following zero–one law. Consider the probability that the statement holds for a structure with universe $\{1, 2, ..., n\}$ and relations chosen at random. This probability approaches either 0 or 1 as n tends to infinity. For numerous examples, see (Blass and Harary, 1979) and the references cited there.

0019-9958/85 $3.00

Glebskii, Kogan, Liogonki, and Talanov (1969) and independently Fagin (1976) showed that every first-order sentence satisfies the zero–one law. Grandjean (1982) showed that the problem of deciding which of the two limit values is correct for a given first-order sentence is PSPACE complete. (We state these results precisely and review their proofs in Sect. 1.) Kaufmann and Shelah have shown that the zero–one law is violated badly within monadic second-order logic.

We extend the zero–one law to sentences in the logic obtained by adding to first-order logic the least-fixed-point operator studied in (Aho and Ullman, 1979; Chandra and Harel, 1982; Immerman, 1982a; Kozen, 1982; Vardi, 1982) or the more powerful iterative fixed point operator (Gurevich, 1984; Livchak, 1983). We show that any formula in these extended logics is equivalent, in random structures (i.e., with probability approaching 1 as the structures get larger), to a first-order formula. This result, which immediately implies the zero–one law, contrasts with the well-known fact that the least-fixed-point operator greatly increases the expressive power of first-order logic.

Contrary to what one might expect, our equivalence result does not allow us to transfer PSPACE completeness of the theory of random structures from first-order logic to the fixed-point operators. The difficulty is that the translation process, from the extended logics to first-order logic, can vastly increase the length of formulas. This difficulty cannot be overcome without proving PSPACE = EXPTIME, for we show that the decision problem for the theory of random structures in logic with a fixed-point operator is EXPTIME hard.

1. The First-Order Theory of Random Structures

In this section, we review for future reference several known theorems, due to Gaifman (1964), Glebskii *et al.* (1969), Fagin (1976), and Grandjean (1982), concerning the first-order theory of random structures with infinite or large finite universes. We present the proofs in a form designed to simplify the proofs of the new results in later sections.

By a *vocabulary* we mean a nonempty finite set of predicate symbols of one or more arguments. We work with a fixed vocabulary σ; by formulas we mean first-order σ-formulas, and by structures we mean σ-structures. A typical example is $\sigma = \{\text{EDGE}\}$, where EDGE is a binary relation symbol, and so structures are directed graphs (possibly with loops).

Let the set ω of natural numbers be made into a σ-structure by means of the following random process. For each predicate symbol P of σ and each tuple (of the appropriate length) $\bar{a}$ of natural numbers, decide whether P

holds of $\bar{a}$ by flipping an unbiased coin. To give a more precise definition of this process, it is convenient to view a σ-structure with universe ω as a function assigning a truth value (0 or 1) to each sentence of the form $P(\bar{a})$ (in the vocabulary σ augmented by names for the natural numbers). Then the process described informally above is defined by the product measure on the space of such functions induced by the probability measure on $\{0, 1\}$ that assigns probability $\frac{1}{2}$ to each element.

THEOREM 1.1. *One isomorphism class contains almost all σ-structures with universe ω. This isomorphism class is the set of countable models of a certain first-order theory* (*explicitly axiomatized below*).

This theorem and an explicit set of axioms for the theory in question were given in Gaifman (1964); the first part of the theorem is also in Erdös and Spencer (1974). The theorem remains correct if the coin used to define the probabilities is biased.

The structures in the isomorphism class given by the theorem will be called *random* structures, and their first-order theory will be called RANDOM (σ). It is easy to see that, if σ' is another vocabulary, then RANDOM (σ) and RANDOM (σ') contain the same sentences of the common vocabulary $\sigma \cap \sigma'$, so it makes sense to say that a sentence is in RANDOM without specifying σ.

Proof Sketch. To describe Gaifman's axiomatization of RANDOM (σ), we first introduce some terminology. For a finite list $\bar{v} = v_1, \dots, v_l$ of distinct variables, a *simple $\bar{v}$-formula* is an atomic formula, with variables from the list $\bar{v}$, and not involving the equality symbol. (Thus, every atomic formula is either a simple $\bar{v}$ or an equation between two variables.) A *complete quantifier-free description for $\bar{v}$*, or just a *$\bar{v}$-description*, is a conjunction of simple $\bar{v}$-formulas and negations of simple $\bar{v}$-formulas such that, for every simple $\bar{v}$-formula α, exactly one of α and $\neg\alpha$ occurs as a conjunct. If w is a variable distinct from the v_i's, then a $\bar{v}$, w-description E *extends* a $\bar{v}$-description D if every conjunct of D is also a conjunct of E. Every such pair D, E (for every $\bar{v}$ and ω) gives rise to one of Gaifman's axioms:

$$\forall \bar{v} \left[\left(\bigwedge_{i<j} v_i \neq v_j \wedge D(\bar{v}) \right) \rightarrow \exists \omega \left(\bigwedge_{i} v^i \neq \omega \wedge E(\bar{v}, \omega) \right) \right].$$

An easy computation shows that every such axiom holds in almost every structure with universe ω. A back-and-forth argument shows that every two countably infinite models of these axioms are isomorphic. These facts suffice to establish Theorem 1.1 once we observe that no finite structure can satisfy all the Gaifman axioms. (We could avoid this observation by adjoining to the Gaifman axioms the sentences which assert the existence of at least n objects, for each n.) ▮

We record, for future reference, two corollaries of the proof of Theorem 1.1.

Corollary 1.2. *The theory* RANDOM (σ) *is* $\aleph_0$*-categorical. (Recall that this means that all models of the theory of cardinality* $\aleph_0$ *are isomorphic.)*

Corollary 1.3. *The theory* RANDOM (σ) *is complete and recursively axiomatized, hence decidable (uniformly in* σ*).*

The decidability result could also be obtained, with a better decision procedure, by an effective eliminination of quantifiers. In fact, Grandjean has shown that we can do considerably better yet.

Theorem 1.4 (Grandjean, 1982). *The decision problem for* RANDOM (σ) *is* PSPACE *complete (with respect to* PTIME *reduction).*

We remark that the degree of the polynomial that bounds the space required by the decision algorithm in (Grandjean, 1982) increases with the arity of the relation symbols in σ. It is thus essential that we are dealing with a fixed finite vocabulary σ.

Proof. We begin with some preliminary facts. First, the same back-and-forth argument as in the proof of Theorem 1.1 shows that, if M is a random structure and $\bar{x}$ and $\bar{y}$ are lists of distinct elements satisfying the same complete quantifier-free description (for a suitable list $\bar{v}$ of variables), then there is an automorphism of M sending $\bar{x}$ to $\bar{y}$. It follows that, if $D(\bar{v})$ is a $\bar{v}$-description and $\bar{x}$ is a tuple of distinct elements satisfying D in some random structure M, then, for any formula $\vartheta(\bar{v})$ with free variables among $\bar{v}$, $\bar{x}$ satisfies ϑ in M if and only if all solutions of D by distinct elements in all random models satisfy ϑ, i.e., if and only if $\bigwedge_{i<j} v_i \neq v_j \wedge D(\bar{v})$ implies $\vartheta(\bar{v})$ in the theory RANDOM. This implies the following equivalences which will be crucial for the correctness of our algorithm; for brevity, we write "$D(\bar{v}) \rightarrow_R \vartheta(\bar{v})$" for "$\forall \bar{v}(\bigwedge_{i<j} v_i \neq v_j \wedge D(\bar{v}) \rightarrow \vartheta(\bar{v}))$ is provable in RANDOM":

$$
\begin{array}{rl}
D(\bar{v}) \underset{R}{\rightarrow} \neg\vartheta(\bar{v}) & \text{if and only if } D(\bar{v}) \underset{R}{\not\rightarrow} \vartheta(\bar{v}) \\[1ex]
D(\bar{v}) \underset{R}{\rightarrow} \vartheta_1(\bar{v}) \vee \vartheta_2(\bar{v}) & \text{if and only if } D(\bar{v}) \underset{R}{\rightarrow} \vartheta_1(\bar{v}) \text{ or } D(\bar{v}) \underset{R}{\rightarrow} \vartheta_2(\bar{v}) \\[1ex]
D(\bar{v}) \underset{R}{\rightarrow} \exists w \vartheta(\bar{v}, w) & \text{if and only if} \\
& \text{either } E(\bar{v}, w) \underset{R}{\rightarrow} \vartheta(\bar{v}, w) \text{ for some extension } E \text{ of } D \\
& \text{or } D(\bar{v}) \underset{R}{\rightarrow} \vartheta(\bar{v}, v_i) \text{ for some } v_i \text{ in the list } \bar{v}.
\end{array}
$$

(The two cases in the last equivalence distinguish whether or not w equals one of the v's.)

The second preliminary fact that we need is a bound on the number of $\bar{v}$-descriptions in terms of the length l of the list $\bar{v}$, the number m of relation symbols in σ, and the maximum arity r of these symbols. Each relation symbol yields, when we assign it arbitrary tuples from $\bar{v}$ as arguments, at most l^r simple $\bar{v}$-formulas. So there are at most $m \cdot l^r$ simple $\bar{v}$-formulas and therefore at most $p(l) = 2^{m \cdot l^r}$ $\bar{v}$-descriptions.

The last preliminary fact is the result of Chandra, Kozen, and Stockmeyer (1981) that PSPACE is equivalent to APTIME, so it suffices to give an alternating PTIME Turing machine to decide membership of sentences in RANDOM (σ). It is convenient to describe first an alternating PTIME algorithm which decides, for any $\bar{v}$, any $\bar{v}$-description $D(\bar{v})$, and any formula $\vartheta(\bar{v})$ with free variables among $\bar{v}$, whether $D \rightarrow_R \vartheta$. We may assume that disjunction and negation are the only connectives and $\exists$ is the only quantifier in ϑ. The algorithm proceeds recursively as follows. If ϑ is $\neg \varphi$, the machine enters a negation state and computes whether $D \rightarrow_R \varphi$. If ϑ is $\vartheta_1 \vee \vartheta_2$, the machine enters an existential state with two successors q_1 and q_2; in q_i it computes whether $D \rightarrow_R \vartheta_i$. If ϑ is $\exists w \varphi(\bar{v}, w)$, the machine enters an existential state with two successors q_1 and q_2. From q_1, it goes through a sequence of $\lceil \log_2 l \rceil$ existential states, guessing an $i \in \{1, ..., l\}$; then it computes whether $D \rightarrow_R \varphi(\bar{v}, v_i)$. From q_2, it goes through a sequence of (at most $m \cdot (l+1)^r$ existential states, guessing an extension $E(\bar{v}, w)$ of $D(\bar{v})$; then it checks whether $E \rightarrow_R \varphi$. If ϑ is a simple $\bar{v}$-formula, it accepts or rejects according to whether ϑ or $\neg \vartheta$ is a conjunct of $D(\bar{v})$. Finally, an equation between variables is accepted if the variables are the same and rejected otherwise.

To decide whether a sentence ϑ is in RANDOM (σ), we apply this algorithm to decide whether TRUE $\rightarrow_R \vartheta$, where TRUE, the empty conjunction, is the unique complete quantifier-free description for the empty list of variables.

Our first preliminary fact shows that this alternating algorithm gives correct answers. Our second preliminary fact shows that it operates in polynomial time, since m and r are fixed (by σ) and l is bounded by the length of the input. Our third preliminary fact follows us to convert the algorithm into a deterministic one that operates in polynomial space. So RANDOM (σ) $\in$ PSPACE.

For the other half of Theorem 1.4, we give a polynomial time computable reduction of the decision problem *QBF* for quantified Boolean formulas to the decision problem for RANDOM (σ). Since *QBF* is known to be PSPACE-complete (Stockmeyer, 1974), this will suffice to complete the proof. To simplify notation, we assume that σ contains a unary predicate symbol P; if it contains only non-unary symbols, just replace variables with sequences of variables in what follows.

The idea of the reduction is simply to let elements satisfying (resp. not satisfying) P act as surrogates in RANDOM (σ) for the truth value 1 (resp. 0) in *QBF*. More precisely, define for each quantified Boolean formula φ a corresponding σ-formula φ' as follows. For a propositional variable p_i, let p_i' be $P(v_i)$. Let $(\neg\varphi)'$ be $\neg(\varphi')$, and similarly for the other connectives. Let $(\exists p_i \psi)'$ be $\exists v_i(\psi')$, and similarly for $\forall$. Then, for every $\varphi(p_1,..., p_l)$ with free propositional variables among $p_1,..., p_l$, φ is true under the truth assignment f: $\{1,..., l\} \to \{0, 1\}$ (i.e., when p_i has value $f(i)$) if and only if the sentence

$$\exists u_0 \exists u_1(\neg P(u_0) \wedge P(u_1) \wedge \varphi'(u_{f(1)},..., u_{f(l)}))$$

is in RANDOM. (The proof of this is a straighforward induction on φ.) In particular, a sentence φ (with no free propositional variables) is in *QBF* if and only if φ' is in RANDOM. This is the desired reduction, so the proof of Theorem 1.4 is complete. ▌

Until now, we have treated only countably infinite structures. However, the theory RANDOM can also be used to provide information about large finite structures, despite the fact that no finite structure can satisfy all of Gaifman's axioms.

For any sentence φ and every positive integer n, let FRACTION (φ, n) be the quotient of the number of models of φ with universe $\{1, 2,..., n\}$ by the total number of σ-structures with this universe. We shall be interested in the behavior of FRACTION (φ, n) as n tends to ∞ (with φ fixed). This behavior indicates the probability that φ is true in a randomly chosen large finite structure.

THEOREM 1.5. (Fagin, 1976; Glebskii, 1969). *If the sentence φ is in* RANDOM (σ), *then* FRACTION $(\varphi, n) \to 1$ as $n \to \infty$. *If φ is not in* RANDOM (σ), *then* FRACTION $(\varphi, n) \to 0$ as $n \to \infty$.

COROLLARY 1.6 (Zero–one law). *For any first-order sentence φ*, $\lim_{n\to\infty}$ FRACTION (φ, n) *exists and equals zero or one.*

Notice that this theorem and corollary would be false if we had permitted zero-place relation symbols in σ, for such a symbol would be a φ with FRACTION $(\varphi, n) = \frac{1}{2}$ for all n. Allowing constant symbols would lead to similar counterexamples.

Sketch of Proof of Theorem 1.5. The second sentence follows from the first, since if RANDOM (σ) does not contain φ it must contain $\neg\varphi$, by completeness (Corollary 1.3). So we need only prove the first assertion. Let φ be a sentence in RANDOM (σ). If φ is one of Gaifman's axioms, then straightforward estimates, which we omit, show that

FRACTION $(\varphi, n) \to 1$. In the general case, φ is a logical consequence of finitely many Gaifman axioms, say $\alpha_1, ..., \alpha_k$. Any structure not satisfying φ must also violate at least one of the α_i's. So

$$1 - \text{FRACTION}\,(\varphi, n) \leqslant \sum_{i=1}^{k} (1 - \text{FRACTION}\,(\alpha_i, n_j)).$$

Since each summand on the right side approaches 0, we see that FRACTION $(\varphi, n) \to 1$ when $n \to \infty$, as desired. ▌

We note for future reference that the last part of the proof actually establishes the general fact that the property "FRACTION $(\varphi, n) \to 1$ as $n \to \infty$" is preserved by logical consequence. We also note that the definition of FRACTION (φ, n) makes sense for any sort of sentence φ (with a well-defined semantics), not just for first-order φ. The preservation of "FRACTION → 1" under logical consequence also continues to hold as long as the number of premises is finite.

2. The Iterative Fixed Point

In this section, we introduce the two extensions of first-order logic that we shall study. These extensions admit formulas defining the least fixed point of a monotone operator or the iterative fixed point of an inflationary operator. Both of these fixed-point constructions have been extensively studied in recursion theory under the names of "monotone" and "non-monotone" induction, respectively; see, for example, (Moschovakis, 1974a, 1974b; Spector, 1961). Much work has been done on logics involving the least-fixed-point operator, sometimes called μ-calculi; see (Aho and Ullman, 1979; de Bakker and de Roever, 1972; Chandra and Harel, 1982; Hitchcock and Park, 1973; Kozen, 1982; Park, 1970; de Roever, 1974; Scott and de Bakker, 1969). The extension of first-order logic by the iterative-fixed-point operator was introduced in Gurevich (1984); a similar concept occurs in Livchak (1983).

Given a structure S of vocabulary σ and a natural number l, consider an operator π which associates with each l-ary predicate P on S a new l-ary predicate $\pi(P)$ on S. Any formula φ of the vocabulary $\sigma \cup \{\dot{P}\}$ (where $\dot{P}$ is a new l-ary predicate symbol), with free variables among $v_1, ..., v_l$, defines such an operator π in every σ-structure S,

$$\pi(P) = \{\bar{x} \in S^l \mid \varphi \text{ holds of } \bar{x} \text{ in the structure } (S, P)\}.$$

If φ has additional free variables, they can be viewed as parameters; an operator π is obtained for any fixed values in S of these parameters. A relation P is a *fixed point* of π if $\pi(P) = P$.

The operator π is *monotone* if, for all l-ary relations P and Q on S, $P \subseteq Q$ implies $\pi(P) \subseteq \pi(Q)$. For monotone operators, a classical construction (Knaster, 1928; Tarski, 1955) provides a least fixed point,

$$\bigcap \{P \subseteq S^l \mid \pi(P) \subseteq P\}.$$

This least fixed point is also obtained by the transfinite inductive construction (Spector, 1961),

$$\begin{aligned} P_0 &= \phi, \\ P_{\alpha+1} &= \pi(P_\alpha). \\ P_\lambda &= \bigcup \{P_\alpha \mid \alpha < \lambda\} \qquad \text{for limit ordinals } \lambda. \end{aligned}$$

It is easy to check that $\alpha < \beta$ implies $P_\alpha \subseteq P_\beta$, so there must be an ordinal number α (of cardinality at most that of S^l) with $P_\alpha = P_{\alpha+1}$. This P_α is the least fixed point of π.

Deciding whether a given first-order formula defines a monotone operator is difficult in general. It is shown in Gurevich (1984) that this problem is (recursively) unsolvable, that it remains unsolvable if one restricts attention to finite structures, and that there is a formula φ such that the problem of deciding whether φ defines a monotone operator on a given finite structure is co-**NP**-complete.

There is, however, a simple syntactic condition on φ that implies monotonicity of the associated operator: $\dot{P}$ should occur only positively in φ (i.e., under an even number of negation symbols). This observation motivates the

LEAST FIXED POINT FORMATION RULE. *Let φ be a formula with only positive occurrences of the l-ary predicate symbol $\dot{P}$, let $v_1,..., v_l$ be distinct variables, and let $u_1,..., u_l$ be variables. Then*

$$(u_1,..., u_l) \in (\mathbf{LFP}\ \dot{P}, v_1,..., v_l)\ \varphi$$

is also a formula.

The vocabulary of the new formula consists of all symbols except $\dot{P}$ from the vocabulary of φ. The free variables of the new formula are $u_1,..., u_l$ and the variables other than $v_1,..., v_l$ that are free in φ. (Thus, **LFP** binds $\dot{P}$, $v_1,..., v_l$.) An occurrence of a predicate symbol (other than $\dot{P}$) in the new formula is positive (resp. negative) if it is so in φ. The new formula is true (in a structure S with specified values for its free variables) if and only if the l-tuple of values of $u_1,..., u_l$ belongs to the least fixed point of the operator defined by φ (and $v_1,..., v_l$, using the specified values for the other variables as parameters). It may be worth remarking that, if we had allowed vocabularies to contain function symbols, then the u's in the new formula would have been allowed to be arbitrary terms and the definitions of vocabulary, free variables, and truth of the new formula would have been modified accordingly.

For non-monotone operators, the sequence of predicates P_α need not stabilize and may therefore yield no fixed point. There is, however, another condition on π that suffices to ensure that $P_\alpha \subseteq P_\beta$ for $\alpha < \beta$ and therefore that $P_\alpha = P_{\alpha+1}$ for some α. This condition is that the operator be *inflationary*, which means that $P \subseteq \pi(P)$ for all P. (The terminology is due, as far as we know, to Freyd (1972) and was first used in the present context in Gurevich, 1984.) Although such an operator need not have a least fixed point, it has a canonically defined fixed point, namely P_α for any sufficiently large α; we call this the *iterative fixed point*.

Deciding whether the operator defined by a formula φ is inflationary is, in general, difficult in the same senses (and by virtually the same proof) as for monotonicity. However, any operator π can be easily transformed into an inflationary operator, $P \mapsto P \cup \pi(P)$, which agrees with π if π happens to be inflationary already. This observation motivates the semantics of the following rule.

Iterative Fixed Point Formation Rule. *Let φ be a formula, P an l-ary predicate symbol* $v_1,..., v_l$ distinct variables, and $u_1,..., u_l$ variables. Then

$$(u_1,..., u_l) \in (\mathbf{IFP}\ \dot{P}, v_1,..., v_l)\ \varphi$$

is also a formula.

The syntactic properties of this new formula are the same as for **LFP**. The formula is true if and only if the l-tuple of values of $u_1,..., u_l$ belongs to the iterative fixed point of $P \mapsto P \cup \pi(P)$, where π is the operator defined by φ. If π is inflationary or monotone, then this is the iterative fixed point of π. In particular, if P occurs only positively in φ, then **IFP** and **LFP** agree, so the logic $FO + \mathbf{IFP}$ obtained by adding the iterative fixed point formation rule to the formation rules of first-order logic subsumes the logic $FO + \mathbf{LFP}$.

It will be useful to have a notation for the stages of the iteration leading to the iterative fixed point.

ITERATION STAGE FORMATION RULE. *For φ, $\dot{P}$, $\bar{v}$, $\bar{u}$ as in the preceding two formation rules and for α an ordinal number*

$$(u_1,..., u_l) \in (\alpha\, \dot{P}, v_1,..., v_l)\, \varphi$$

is a formula.

The syntactic properties of the new formula are as for the preceding two rules; truth is defined as for **IFP** but with "the αth stage P_α" in place of "the iterative fixed point." We shall need this formation rule only for finite α, and we adopt for complexity-theoretic purposes the convention that the new formula should contain α written in binary notation.

For finite α, α steps in the iteration of a first-order definable operator can be carried out in first-order logic. Thus, the new formula $\bar{u} \in (\alpha\dot{P}, \bar{v})\, \varphi$ is equivalent to a first-order formula if φ is, so the iteration stage formation rule adds no expressive power to first-order logic unless α is infinite. Indeed, $\bar{u} \in (\mathbf{O}\, \dot{P}, \bar{v})\, \varphi$ is always false, and $\bar{u} \in (\mathbf{k+1}\, \dot{P}, \bar{v})\, \varphi$ is equivalent to the result of first substituting $\bar{u}$ for $\bar{v}$ in φ (renaming bound variables if necessary), then replacing every subformula of the form $\dot{P}(\bar{w})$ (for arbitrary $\bar{w}$) with $\bar{w} \in (\mathbf{k}\, \dot{P}, \bar{v})\, \varphi$, and finally forming the disjunction of the result with $\bar{u} \in (\mathbf{k}\, \dot{P}, \bar{v})\, \varphi$. Recursive application of this procedure lets us reduce any formula in first-order logic with the iteration stage rule for finite α, $FO + \mathbf{IS}$, to a first-order formula. It should be noted, however, that this translation process may result in a formula vastly longer than the initial formula. Thus, although the iteration stage rule for finite α does not increase expressive power, it does affect complexity.

3. THE ZERO–ONE LAW FOR FIRST-ORDER LOGIC WITH ITERATIVE FIXED POINT

The purpose of this section is to extend the zero-one law, Corollary 1.6, from first-order logic to the stronger logic $FO + \mathbf{IFP}$ introduced in Section 2.

THEOREM 3.1. *Let φ be a formula of $FO + \mathbf{IFP}$ with vocabulary σ. There exist a first-order σ-formula φ' and a finite subset G of Gaifman's axiom set for* RANDOM (σ) *such that φ and φ' are equivalent in all models of G.*

COROLLARY 3.2 (Zero–one law). *Let φ be a sentence of $FO + \mathbf{IFP}$. Then* $\lim_{n \to \infty}$ FRACTION (φ, n) *exists and equals zero or one.*[1]

[1] Immerman has written to us that the zero–one law also follows from his work (Immerman, 1982b) on expressibility with a bounded number of variables. The law isn't there in any explicit form.

Proof of Corollary 3.2. Let φ' and G be as in the theorem. Since each sentence in G has FRACTION → 1, and since either FRACTION $(\varphi', n) \to 1$ or FRACTION $(\neg\varphi', n) \to 1$ by Corollary 1.6, we have that either φ, which is a logical consequence of $G \cup \{\varphi'\}$, or $\neg\varphi$, which is a logical consequence of $G \cup \{\neg\varphi'\}$, has FRACTION → 1. ▌

Proof of Theorem 3.1. Let σ be the vocabulary of φ. Since the theory RANDOM (σ) is $\aleph_0$-categorical, we need only invoke Theorem 1 of Appendix 3 of Gurevich (1984). For the sake of completeness, we sketch the proof.

We proceed by induction on the depth of nesting of **IFP** in φ. The only nontrivial case is that φ is $\bar{u} \in (\mathbf{IFP}\, \dot{P}, \bar{v})\, \psi$. We cannot apply the induction hypothesis to ψ since ψ involves $\dot{P}$ (whose interpretation is certainly not random here), but we can apply it to any finite stage in the iteration of ψ. More precisely, define, for each natural number k, a first-order formula $\psi_k(\bar{u})$ that does not contain $\dot{P}$ and is equivalent to $\bar{u} \in (\mathbf{k}\, \dot{P}, \bar{v})\, \psi$ in all models of a certain finite subset G_k of Gaifman's axioms. For $k = 0$, let $\psi_0(\bar{u})$ be FALSE. Let $\psi_{k+1}(\bar{u})$ be obtained from ψ by first substituting $\bar{u}$ for $\bar{v}$, then replacing every subformula of the form $\dot{P}(\bar{w})$ with $\psi_k(\bar{w})$, then applying the induction hypothesis to get an almost equivalent first-order formula, and finally forming the disjunction with $\psi_k(\bar{u})$. Here "almost equivalent" means "equivalent in all models of enough of the Gaifman axioms." Comparison with the discussion at the end of Section 2 shows that the ψ_k's have the desired properties.

In a countable model M of RANDOM (σ), each $\psi_k(\bar{u})$ is equivalent to $\bar{u} \in (\mathbf{k}\, \dot{P}, \bar{v})\, \psi$, since all of Gaifman's axioms hold. In particular, we have the monotonicity property that each ψ_k implies the next, ψ_{k+1}, in M. Since RANDOM (σ) is $\aleph_0$-categorical, a version of Ryll–Nardzewski's theorem (Theorem 2.3.12(e) in Chang and Keisler, 1973) asserts that there are only finitely many inequivalent (in M) formulas with a fixed finite set of free variables; in particular, some ψ_k and ψ_l with $k < l$ must be equivalent in M. Then ψ_k is equivalent to ψ_{k+1} in M, because of the monotonicity property. This equivalence, being a first-order statement true in M, is deducible from finitely many Gaifman axioms. In any model of these finitely many axioms plus the finitely many more needed to ensure that $\psi_k(\bar{u})$ and $\psi_{k+1}(\bar{u})$ are equivalent to $u \in (\mathbf{k}\, \dot{P}, \bar{v})\, \psi$ and $\bar{u} \in (\mathbf{k+1}\, \dot{P}, \bar{v})\, \psi$ respectively, all these equivalences together ensure that the iteration defining φ stops after the kth step and that φ is equivalent to ψ_k. This completes the proof of Theorem 3.1. ▌

We have presented this proof in a form applicable to any $\aleph_0$-categorical theory. For the particular theory RANDOM (σ), we can obtain an improvement, which will be useful in Section 4, by replacing the use of Ryll–Nardzewski's theorem with the more explicit bounds obtained, as the second preliminary fact, in the proof of Theorem 1.4. In that proof, we saw that there are at most $2^{m \cdot l^r}$ $\bar{v}$-descriptions, where l is the length of $\bar{v}$ and m and r are determined by σ (the number of predicate symbols and the maximum arity). Since tuples satisfying the same complete quantifier-free description are related by an automorphism of M (the first preliminary fact in the proof of Theorem 1.4), they satisfy the same formulas of $FO + \mathbf{IFP}$. It follows that, as k increases, the sequence of predicates defined by $\bar{u} \in (\mathbf{k}\ \dot{P}, \bar{v})\psi$ cannot strictly increase more than $p(l) = 2^{m \cdot l^r}$ times, where l is the number of free variables of ψ. Thus, in $\bar{u} \in (\mathbf{IFP}\dot{P}, \bar{v})\psi$, we can replace **IFP** by $p(l)$ (or any larger number). Doing this systematically, we can transform any $FO + \mathbf{IFP}$ formula φ into an equivalent (in M) formula of $FO + \mathbf{IS}$ in which the iteration stage formation rule is applied only with $\alpha = p(l)$, where l is the number of variables in φ. As we saw at the end of Section 2, this $FO + \mathbf{IS}$ formula is equivalent (in all structures) to a first-order formula φ''. Since this φ'' and the φ' obtained in the proof of Theorem 3.1 are equivalent in the countable models of RANDOM (σ), they are also, by compactness, equivalent in all models of a certain finite set of Gaifman axioms. Therefore, φ'' has the property asserted of φ' in Theorem 3.1.

The following proposition summarizes this discussion for future reference.

PROPOSITION 3.3. *The φ' in Theorem 3.1. can be taken to be the first-order translation of a formula of $FO + \mathbf{IS}$ in which the stages mentioned are all $2^{m \cdot l^r}$, where l is the number of variables in φ.*

4. THE COMPLEXITY OF THE FO + IFP THEORY OF RANDOM STRUCTURES

The proofs of the zero-one laws for first-order logic and for $FO + \mathbf{IFP}$ show that a sentence is true in random (countably infinite) structures if and only if it is true in almost all finite structures in the sense that FRACTION $(\varphi, n) \to 1$ as $n \to \infty$. We say that a sentence with these equivalent properties is *almost surely true.* This section is devoted to determining the complexity of the decision problem for almost sure truth of sentences in $FO + \mathbf{LFP}$ and $FO + \mathbf{IFP}$. Recall that for first-order logic this problem is PSPACE complete (Theorem 1.4).

THEOREM 4.1. *The decision problem for almost sure truth of $FO + \mathbf{IFP}$ sentences can be solved by an alternating Turing machine in polynomial space.*

Proof. The machine proceeds according to the following algorithm.

Given an $FO + \mathbf{IFP}$ sentence φ with l variables, it replaces every occurrence of **IFP** with p where $p = p(l) = 2^{m \cdot l^r}$ and m and r are, as before, the number and the maximum arity of predicate symbols in σ. By Proposition 3.3, this replacement almost surely does not alter the truth value of φ. Since p is written in binary notation, and since $l <$ length of φ, the space required is polynomial in the length of φ.

The rest of the algorithm is exactly like that in the proof of Theorem 1.4, with the following additional steps to handle the iteration stage formation rule. To decide whether $D \rightarrow_R \bar{u} \in (\mathbf{k} + 1\dot{P}, \bar{v})\, \psi$, where D is a complete quantifier-free description for appropriate variables, decide instead whether $D \rightarrow_R \vartheta$ where ϑ is obtained from ψ by substituting $\bar{u}$ for $\bar{v}$, replacing every $\dot{P}(\bar{w})$ with $\bar{w} \in (\mathbf{k}\, \dot{P}, \bar{v})\, \psi$, and forming the disjunction with $\bar{u} \in (\mathbf{k}\, \dot{P}, \bar{v})\, \psi$. (See the end of Sect. 2). Finally, to decide whether $D \rightarrow_R \bar{u} \in (\mathbf{O}\, \dot{P}, \bar{v})\, \psi$, reject. The algorithm never needs to deal with complete quantifier-free descriptions for more than l variables, so every description it uses is a conjunction of at most $m \cdot l^r$ simple formulas and negations of simple formulas. Thus, the machine needs only polynomial space to record descriptions and iteration stages. Its other storage needs are comparatively minor, so it operates in polynomial space, and the theorem is proved. ∎

In the following corollary, EXPTIME refers to deterministic Turing computation with a time bound $2^{f(n)}$, where f is a polynomial (not necessarily linear) function of the input length n.

COROLLARY 4.2. *Almost sure truth of $FO + \mathbf{IFP}$ sentences is decidable in* EXPTIME.

Proof. EXPTIME is equivalent to alternating PSPACE, by Chandra, Kozen, and Stockmeyer (1981). ∎

THEOREM 4.3. *The decision problem for almost sure truth of $FO + \mathbf{LFP}$ sentences is* EXPTIME *hard.*

Proof. Because of the result of Chandra *et al.* (1981) just quoted, it suffices to reduce, to the decision problem for almost sure truth of $FO + \mathbf{LFP}$ sentences, every language recognized by an alternating Turing machine that operates in polynomial space. For simplicity, we assume that our alternating machines have only universal and existential states, not negation states; it is shown in Chandra *et al.* (1981) this assumption causes no loss of generality. Let M be such a machine, and let $S(|w|)$ be a polynomial in the length of its input w that bounds the space used by M. We show how to compute, in polynomial time, from any given input w a sentence ϑ_w in $FO + \mathbf{LFP}$ such that ϑ_w is almost surely true if and only if M accepts w.

To construct ϑ_w, we use the well-known fact (Cook, 1971) that the activity of a Turing machine can be described by a string of truth values (or bits). In the present situation, the computation of M on input w is too large to be useful for our purposes, but each configuration (instantaneous description) of M can be coded by a string of length polynomial in $|w|$. For concreteness, we adopt the convention that, if M has s states and a tape symbols, then the bit strings have length $s+(a+1)\cdot S(|w|)$ and consist of the truth values of the following statements about the configuration: "M is in state q" for each of the s states q, "M is scanning square n" for each of the $S(|w|)$ relevant squares, and "the symbol in square n is Z" for each of the $S(|w|)$ relevant squares and each of the α symbols Z. Of course, a string corresponds to a configuration only if it satisfies some obvious consistency conditions: M is in exactly one state, scanning exactly one square, and each square has exactly one symbol in it (when the blank counts as a symbol).

As in the proof of Theorem 1.4, we may assume for notational simplicity that σ contains a unary predicate symbol Q. For each input word w, we can easily construct, in polynomial time, first-order formulas $\mathrm{INITIAL}_w(\bar{x})$, $\mathrm{UNIVERSAL}_w(\bar{x})$, $\mathrm{EXISTENTIAL}_w(\bar{x})$, $\mathrm{YES}_w(\bar{x})$, and $\mathrm{SUCCESSOR}_w(\bar{x}, \bar{y})$, in which $\bar{x}$ and $\bar{y}$ are sequences of $l=s+(a+1)\cdot S(|w|)$ variables, and which assert, respectively, that the string of bits $\bar{Q}(\bar{x})=Q(x_1)\cdots Q(x_l)$ codes the initial configuration with input w, that $\bar{Q}(\bar{x})$ codes a configuration where M is in a universal state, that $\bar{Q}(\bar{x})$ codes a configuration where M is in an existential state, that $\bar{Q}(\bar{x})$ codes a configuration where M is in an accepting (terminal) state, and that M can go in one computation step from the configuration coded by $\bar{Q}(\bar{x})$ to that coded by $\bar{Q}(\bar{y})$. Of these five formulas, the last four depend on w only through the dependence of l on the length of w.

Recall that, by the definition of the way alternating Turing machines operate, the set of configurations that M accepts is the smallest set A such that (i) A contains every configuration in which M is in an accepting terminal state, (ii) A contains every universal configuration all of whose successors are in A, and (iii) A contains every existential configuration at least one of whose successors is in A. This means that the sets of $\bar{x}$ for which $\bar{Q}(\bar{x})$ codes an accepting configuration is definable by

$$\mathrm{ACCEPT}_w(\bar{x}) \leftrightarrow \bar{x} \in (\mathbf{LFP}\ \dot{P}, \bar{v})\ \varphi$$

where φ is

$$\begin{aligned}\mathrm{YES}_w(\bar{v}) &\vee (\mathrm{UNIVERSAL}_w(\bar{v}) \wedge \forall \bar{z}(\mathrm{SUCCESSOR}_w(\bar{v}, \bar{z}) \rightarrow \dot{P}(\bar{z}))\\ &\vee (\mathrm{EXISTENTIAL}_w(\bar{v}) \wedge \exists \bar{z}(\mathrm{SUCCESSOR}_w(\bar{v}, \bar{z}) \wedge \dot{P}(\bar{z})).\end{aligned}$$

This definition works in any structure where at least one element satisfies Q and at least one does not, so that every code occurs as $\bar{Q}(\bar{z})$ for some z; no other properties of random structures are needed for this proof.

Finally, we have that M accepts w if and only if the following sentence ϑ_w is almost surely true:

$$\exists\bar{x}(\mathrm{INITIAL}_w(\bar{x}) \wedge \mathrm{ACCEPT}_w(\bar{x})).$$

Since ϑ_w can clearly be written down in polynomial time when w is given, Theorem 4.3 is proved. ▌

Corollary 4.4. *The decision problems for almost sure truth in FO* + **LFP** *and FO* + **IFP** *are* EXPTIME *complete.*

Proof. Combine Corollary 4.2, Theorem 4.3, and the fact that **IFP** subsumes **LFP**. ▌

5. Unbounded Vocabulary

In the preceding sections, we have worked with a fixed finite vocabulary σ. The number m of relation symbols in σ and the maximum arity r of these symbols played an important role in the proof of Theorem 4.1, where we constructed an alternating Turing machine that decides in space roughly propotional to $m \cdot l^r$, whether a *FO* + **IFP** sentence φ with l variables is almost surely true. If we remove the restriction to a fixed vocabulary σ, then m and r are no longer constant. They are, of course, majorized by the length of φ, but the space bound so obtained for our algorithm is exponential, rather than polynomial, in the length of φ because r occurs as an exponent. (If σ is allowed to vary with r bounded, then the complexity estimate of Theorem 4.1 remains correct.) Thus, the method of Theorem 4.1. gives only the following upper bound for the complexity of the *FO* + **IFP** theory of random structures, with no restrictions on the vocabulary σ.

Theorem 5.1. *The FO* + **IFP** *theory of random structures is decidable in alternating exponential space.*

By Chandra *et al.* (1981), alternating exponential space is equivalent to deterministic double-exponential time.

The purpose of this section is to prove that Theorem 5.1 is optimal, i.e., that RANDOM is complete for alternating exponential space. The proof is similar to that of Theorem 4.3, the differences being that the space bound S is now exponential rather than polynomial and that we can (and must) use relations of high arity for the coding of machine configurations.

THEOREM 5.2. *The decision problem for* RANDOM *in* $FO+\mathbf{LFP}$ *is hard, with respect to polynomial time reductions, for alternating exponential space.*

Sketch of Proof. As in the proof of Theorem 4.3, let M be an alternating Turing machine with only universal and existential states, and let the space it needs, for input w, be bounded by $S(|w|)$, where now $S(n)=2^{p(n)}$ and p is a polynomial. To code configurations of M, we use s unary relations STATE q, one for each state q of M, $a\cdot p(|w|)$-ary relations CELL z, one for each tape symbol z of M including the blank symbol O, and one additional $p(|w|)$-ary relation HEAD. We now define what it means for a pair of distinct elements x, y in a structure for this vocabulary to code a configuration C of M. In the definition, we let $\bar{t}$ stand for a tuple of length $p(|w|)$ each of whose components is x or y; the $2^{p(|w|)}=S(|w|)$ such tuples are lexicographically ordered (with x preceding y), and the ith tuple in this ordering is to be considered a name of the ith tape cell of M. Then (x, y) codes C if, for all q, z, $\bar{t}$ as above,

(1) STATE $q(x)$ holds if and only if q is the state in C,

(2) HEAD$(\bar{t})$ holds if and only if $\bar{t}$ names the square scanned in C, and

(3) CELL $z(\bar{t})$ holds if and only if the square named by $\bar{t}$ contains the symbol z in C.

Of course, a pair (x, y) codes a configuration only if the truth values in (1), (2), and (3) satisfy appropriate consistency conditions. We use randomness to ensure (via finitely many Gaifman axioms) that every configuration has a code.

Pairs (x, y) will play the same role in the present proof that tuples $\bar{x}$ played in the proof of Theorem 4.3. We shall define INITIAL$_w(x, y)$, UNIVERSAL$_w(x, y)$, EXISTENTIAL$_w(x, y)$, YES$_w(x, y)$, and SUCCESSOR$_w(x, y, x', y')$ with the same meanings as the formulas with the same names in the proof of 4.3; once this is done, the construction of ϑ_w is exactly as before.

The formulas INITIAL$_w(x, y)$, etc., can be produced by the well-known methods of Cook (1971) once we have formulas describing the way tape squares are named. We shall exhibit formulas NAME$_w(x, y, \bar{u})$, $<_w(x, y, \bar{u}, \bar{v})$, NEXT$_w(x, y, \bar{u}, \bar{v})$, and FIRST$_w(x, y, \bar{u})$ which assert that, with respect to x and y, $\bar{u}$ names a tape square, the square named by $\bar{u}$ is to the left of that named by $\bar{v}$, the square named by $\bar{u}$ is immediately to the left of that named by $\bar{v}$, and $\bar{u}$ names the leftmost square, respectively. In all these formulas, $\bar{u}$ and $\bar{v}$ are $p(|w|)$-tuples of variables. For readability, we shall suppress the subscript w and the arguments x, y, we shall write $<$

between its arguments, and we adopt the convention that i and j range from 1 to $p(|w|)$.

$$\mathrm{NAME}(\bar{u}) \leftrightarrow \bigwedge_i (u_i = x \vee u_i = y)$$

$$\bar{u} < \bar{v} \leftrightarrow \bigvee_i (u_i = x \wedge v_i = y \wedge \bigwedge_{j<i} u_j = v_j)$$

$$\mathrm{NEXT}(\bar{u}, \bar{v}) \leftrightarrow \bar{u} < \bar{v} \wedge \neg\exists\bar{t}(\mathrm{NAME}(\bar{t}) \wedge \bar{u} < \bar{t} \wedge \bar{t} < \bar{v})$$

$$\mathrm{FIRST}(\bar{u}) \leftrightarrow \bigwedge_i u_i = x.$$

Now we are in a position to define INITIAL, UNIVERSAL, EXISTENTIAL, YES, and SUCCESSOR. Since the ideas involved are quite standard, we define INITIAL as an example and leave the rest to the reader. In this definition, q ranges over the states of M, q_0 is the initial state, k ranges from 1 to $|w|$, w_k is the kth letter in the word w, O is the blank symbol, and $\bar{v}, \bar{u}^1, ..., \bar{u}^{|w|}$ are $p(|w|)$-tuples of variables distinct from each other and from x and y:

$$\begin{aligned}
&\mathrm{INITIAL}_w(x, y) \leftrightarrow \mathrm{STATE}\, q_0(x) \wedge \bigwedge_{q \neq q_0} \neg \mathrm{STATE}\, q(x) \\
&\wedge \exists\bar{u}^1 \cdots \exists\bar{u}^{|w|}(\mathrm{FIRST}(\bar{u}^1) \wedge \forall\bar{v}((\mathrm{HEAD}(\bar{v}) \wedge \mathrm{NAME}(\bar{v})) \leftrightarrow \bar{v} = \bar{u}^1) \\
&\bigwedge_{k<|w|} (\mathrm{NAME}(\bar{u}^{k+1}) \wedge \mathrm{NEXT}(\bar{u}^k, \bar{u}^{k+1})) \wedge \bigwedge_k \mathrm{CELL}\, w_k(\bar{u}^k) \\
&\wedge \forall\bar{v}((\mathrm{NAME}(\bar{v}) \wedge \bigwedge_k \neg(\bar{v} = \bar{u}^k)) \rightarrow \mathrm{CELL}\, O(\bar{v})).
\end{aligned}$$

UNIVERSAL, EXISTENTIAL, and YES are much easier to define; they refer only to STATE q predicates. SUCCESSOR is more tedious but straightforward. Once these definitions are available, we can produce ϑ_w as in the proof of Theorem 4.3, thereby completing the present proof. ∎

The coding technique used in the preceding proof and the technique used in the well-known proof (Hopcroft and Ullman, 1979) of Stockmeyer's theorem that *QBF* is PSPACE-complete can be combined to show that the decision problem for the first-order theory RANDOM is EXPTIME-hard. This decision problem is solvable in AEXPTIME = EXSPACE by means of the algorithm in the proof of Grandjean's Theorem 1.4 above. In response to an earlier version of this paper, Immerman informed us that this decision problem is complete for alternating exponential time with polynomially many alternations.

6. Fixed-Points of Bounded Arity

In this section, we reinstate the assumption that we are working with a fixed finite vocabulary σ, and, in addition, we restrict the arity of the predicates that we allow **IFP** or **LFP** to define. More precisely, let $FO+\mathbf{IFP}_k$, where k is a nonnegative integer, be the logic obtained by adding to first-order logic the **IFP** formation rule subject to the constraint that **IFP** can be applied only to formulas with at most k free variables. We show that the complexity of the decision problem for almost sure truth in this logic is, for each fixed k, the same (modulo PTIME reductions) as in first-order logic.

THEOREM 6.1. *The decision problem for almost sure truth of* $FO+\mathbf{IFP}_k$-*sentences is, for each fixed* k, PSPACE-*complete.*

Proof. In view of Theorem 1.4, it suffices to give an algorithm for solving the decision problem in polynomial space. As in Section 4, we proceed by extending the algorithm in the proof of Theorem 1.4 to cover formulas involving **IFP**. This time, however, it will be convenient to work with the deterministic polynomial space version of the alternating polynomial time algorithm of Theorem 1.4. This deterministic version, as constructed in Chandra *et al.* (1981), is essentially a systematic depth-first search of the computation tree of the alternating algorithm; its space requirements are only polynomial because it keeps track of only the choices made by the alternating machine on the branch leading to the node currently being simulated and it re-uses the space previously used for computations from other branches.

To handle **IFP**, we expand the algorithm as follows. Before beginning its actual computation, the machine writes, on a portion of the tape that will not be needed otherwise, a list of all complete quantifier-free descriptions $D(\bar{v})$ for some lists $\bar{v}$ of variables, one list of each length $\leqslant k$. It leaves a little space after each $D(\bar{v})$ to allow descriptions to be marked later in the algorithm; this marking space after each $D(\bar{v})$ is to consist of as many tape squares as the maximum depth of nested **IFP**'s in the formula φ to be tested. Since the number of $\bar{v}$-descriptions is at most $\sum_{l\leqslant k} 2^{m\cdot l^r}$, independently of φ, the space used here is linear in $|\varphi|$.

After these preparatory steps, the algorithm begins to operate like the deterministic version of the one in Theorem 1.4. When it encounters an IFP operator, of depth d (measured from the outside), it proceeds to mark (in the dth square of each space provided for this purpose) those $\bar{v}$-descriptions $D(\bar{v})$ such that the tuples satisfying $D(\bar{v})$ also satisfy the predicate defined by this **IFP**. It does this by starting with all $\bar{v}$-descriptions unmarked in the dth space, erasing, if necessary, any marks already there (corresponding to $P_0=\phi$ in the definition of iteration stages) and following the definition of the stages P_n to mark $D(\bar{v})$'s as the tuples satisfying them enter the predicate being inductively defined. At the nth stage, the descriptions of tuples in P_{n-1} are already marked, and the algorithm evaluates the formula ψ to which **IFP** was applied, for tuples satisfying the remaining descriptions, using the currently marked descriptions to interpret $\dot{P}$. Any tuples found to satisfy ψ are marked at the next stage, for they belong to P_n. The evaluation of ψ may, of course, involve further uses of this marking procedure, if ψ involves **IFP**'s. The maximum number of marking processes that ever proceed simultaneously during execution of the algorithm equals the maximum nesting of **IFP**'s in φ, which is why we provided this much space for markings. ▌

Note that it was essential to this proof that not only the number of free variables in the scope of an **IFP** be bounded (by k) but also that the vocabulary be fixed so that m and r are constants. With unbounded vocabulary, replacing **IFP** with $\mathbf{IFP}_k$ does not improve the complexity estimates. To see this, simply observe that, in the proof of Theorem 5.2, **IFP** was applied only to a formula with just two variables.

The restriction on the **IFP** formation rule in $FO+\mathbf{IFP}_k$ bounds not only the number of variables bound by **IFP** but also the number of additional variables (parameters) in the formula to which **IFP** is applied. We do not know the complexity of the decision problem for almost sure truth in a logic where only the number of variables quantified by **IFP** is bounded while the number parameters is unrestricted.

7. Additional Remarks

We pointed out in the proof of Theorem 4.3 that the coding of alternating Turing machines used there can be done in any structure where at least one element satisfies Q and at least one element does not satisfy Q; no further use is made of randomness. In particular, we could take the structure to be the two-element Boolean algebra and take $Q(x)$ to be $x=1$. Thus, if we extend the theory QBF of quantified Boolean formulas by adjoining **LFP** to the logic, the resulting theory is EXPTIME hard. In fact, so is the $FO+\mathbf{LFP}$ theory of any structure with more than one element, since we can use the binary predicate of equality in place of the unary predicate Q. It is easy to check that the $FO+\mathbf{LFP}$ and $FO+\mathbf{IFP}$ theories of the two-element Boolean algebra or of any non-trivial set (with only the equality predicate) are decidable in EXPTIME and are therefore complete for this class.

The results proved in this paper for general structures also hold for certain restricted classes of structures, for example undirected graphs (=irreflexive symmetric binary relations). A zero-one law for the first-order theory of almost all structures in a first-order definable class can be transferred to the $FO+\mathbf{IFP}$ theory by our methods provided the almost surely true first-order sentences constitute an $\aleph_0$-categorical theory. If we have, in addition, effective estimates for the number of inequivalent types of l-tuples (a number that is finite for each l by Ryll-Nardzewski's theorem) and effective ways of describing these types, then our methods also provide upper bounds on the complexity of the decision problem for almost sure truth. All of these apparently stringent hypotheses are satisfied in the case of undirected graphs and in the case of simplicial complexes (of arbitrary but fixed dimension).

Although we worked with finite structures with a fixed universe $\{1, 2, ..., n\}$ (labeled structures), our results apply also to isomorphism classes (unlabeled structures). Indeed, if FRACTION(φ, n) were defined using numbers of isomorphism classes in both the numerator and the denominator, the new numerator and denominator would be asymptotically equal to the old divided by $n!$ and the value of FRACTION would thus be asymptotically unchanged, because almost all structures have no non-trivial automorphisms.

RECEIVED: September 24, 1984; ACCEPTED: July 15, 1985

REFERENCES

AHO, A. AND ULLMAN, J. (1979), Universality of data retrieval languages, *in* "Proc. 6th ACM Sympos. on Principles of Programming Languages," 110–120.

BLASS, A. AND HARARY, F. (1979), Properties of almost all graphs and complexes, *J. Graph Theory* **3**, 225–240.

DEBAKKER, J. W. AND DEROEVER, W. (1972), A calculus for recursive program schemes, *in* 1st Internat. Colloq. on Automata, Languages and Programming, 167–196, North-Holland, Amsterdam.

CHANDRA, A. AND HAREL, D. (1982), Structure and complexity of relational queries, *J. Comput. System Sci.* **25**, 99–128.

CHANDRA, A., KOZEN, D., AND STOCKMEYER, L. (1981), Alternation, *J. Assoc. Comput. Mach.* **28**, 114–133.

CHANG, C. C. AND KEISLER, H. J. (1973), "Model Theory," North-Holland, Amsterdam.

COOK, S. (1971), The complexity of theorem-proving procedures, *in* "Proc. 3rd ACM Sympos. on Theory of Computing," pp. 151–158.

ERDÖS, P. AND SPENCER, J. (1974), "Probabilistic Methods in Combinatorics," Academic Press, New York.

FAGIN, R. (1976), Probabilities on finite models, *J. Symbolic Logic* **41**, 50–58.

FREYD, P. (1972), Aspects of topoi, *Bull, Austral. Math. Soc.* **7** 1–76.

GAIFMAN, H. (1964), Concerning measures in first-order calculi, *Israel J. Math.* **2**, 1–18.

GLEBSKII, Y. V., KOGAN, D. I., LIOGONKI, I. M., AND TALANOV, V. A. (1969), The extent and degree of satisfiability of a form of the retricted predicate calculus, *Kibernetika* **2**, 31–42.

GRANDJEAN, E. (1982), Complexity of the first theory of almost all finite structure, preprint; "Logique des Structures Finies et Complexité Algorithmique," thesis, Université Lyon I, pp. 11–47, 1984.

GUREVICH, Y. (1984), Toward logic tailored for computational complexity, *in* "Computation and Proof Theory" (M. M. Richter *et al.*, Eds.), pp. 175–216, Springer Lecture Notes in Math. 1104.

HITCHCOCK, P. AND PARK, D. M. R. (1973), Induction rules and termination proofs, *in* "Proc. 1st Internat. Colloq. on Automata, Languages, and Programming, pp. 225–251, North-Holland, Amsterdam.

HOPCROFT, J. E. AND ULLMAN, J. D. (1979), "Introduction to Automata Theory, Languages and Computation," Addison–Wesley, Reading, Mass.

IMMERMAN, N. (1982a), Relational queries computable in polynomial time, *in* "Proc. 14th ACM Symposium on Theory of Computing," pp. 147–152.

IMMERMAN, N. (1982b), Upper and lower bounds for first order expressibility, *J. Comput. System Sci.* **25**, 76–98.

KNASTER, B. (1928), Un théorème sur les fonctions d'ensembles, *Ann. Soc. Polon. Math.* **6**, 133–134.

KOZEN, D. (1982), Results on the propositional μ-calculus, *in* "Proc. 9th Internat. Colloq. on Automata, Languages, and Programming," pp. 348–369.

KAUFMANN, M. AND SHELAH, S. (to appear), On random models of finite power and monadic logic, *Discrete Math.*

LIVCHAK, A. B. (1983), The relational model for process control, *Automat. Document. Math. Linguistics* **4**, 27–29. [Russian]

MOSCHOVAKIS, Y. (1974), Elementary Induction on Abstract Structures," North-Holland, Amsterdam.

MOSCHOVAKIS, Y. (1974b), On non-monotone inductive definability, *Fund. Math.* **82**, 39–83.

PARK, D. M. R. (1970), Fixpoint induction and proof of program semantics, *in* "Machine Intelligence" Vol. 5 (B. Meltzer and D. Michie, Eds.), Edinburgh Univ. Press, Edinburgh.

DEROEVER, W. P. (1974), "Recursive Program Schemes: Semantics and Proof Theory," Ph. D. thesis, Free University, Amsterdam.

SCOTT, D. AND DEBAKKER, J. W. (1968), A theory of programs, unpublished manuscript, IBM, Vienna.

SPECTOR, C. (1961), Inductively defined sets of natural numbers, *in* "Infinitistic Methods," Proc., Sympos. on Foundations of Math., Warsaw, 1959, pp. 97–102, Pergamon, Oxford.

STOCKMEYER, L. (1974), "The Complexity of Decision Problems in Automata Theory and Logic," MAC-TR-133, Project MAC, MIT, Cambridge, Mass.

TARSKI, A. (1955), A lattice-theoretical fixpoint theorem and its applications, *Pacific J. Math.* **5**, 285–309.

VARDI, M. (1982), Complexity of relational query languages, *in* "Proc. 14th ACM Symposium on Theory of Computing," pp. 137–146.