JOURNAL OF COMPUTER AND SYSTEM SCIENCES 50, 493–505 (1995)

Descriptive Complexity of #P Functions

SANJEEV SALUJA* AND K. V. SUBRAHMANYAM

Department of Computer Science, Tata Institute of Fundamental Research, Bombay 400 005, India

AND

MADHUKAR N. THAKUR†

Computer and Information Sciences, University of California, Santa Cruz, California 95064

Received February 3, 1994

We give a logic-based framework for defining counting problems and show that it exactly captures the problems in Valiant's counting class #P. We study the expressive power of the framework under natural syntactic restrictions and show that some of the subclasses obtained in this way contain problems in #P with interesting computational properties. In particular, using syntactic conditions, we isolate a class of polynomial time computable #P problems and, also, a class in which every problem is approximable by a polynomial time randomized algorithm. These results set the foundation for further study of the descriptive complexity of the class #P. In contrast, we show, under reasonable complexity theoretic assumptions, that it is an undecidable problem to tell if a counting problem expressed in our framework is polynomial time computable or if it is approximable by a randomized polynomial time algorithm. Finally, we discuss some open problems which arise naturally from this work.

1. INTRODUCTION

In computer science, computational models like Turing machines have been well studied as recognizers of languages (subsets of $\{0, 1\}^*$). This is especially true in complexity theory where resources like time and space can be defined naturally using these machine models and the resource complexity of recognizing languages can be studied. On the other hand, if strings in $\{0, 1\}^*$ are viewed as encodings of finite structures over an appropriate vocabulary, then formulae in a certain logic can be viewed as recognizers of languages in the following way: Let σ be a vocabulary and let ϕ be a sentence in a certain logic. One can associate with the sentence ϕ the language $\{\mathbf{A} : \mathbf{A} \models \phi\}$ or more formally, the language $\{e(\mathbf{A}) \in \{0, 1\}^* : \mathbf{A} \models \phi\}$, where $\mathbf{A}$ is a finite structure over the vocabulary σ and $e(\mathbf{A})$ is an encoding of the structure $\mathbf{A}$ as a string in a natural way.

Such a logical framework for defining languages becomes much more interesting to a complexity theorist if there are appropriate logics that capture exactly the languages in the complexity classes that are otherwise defined using computational machine models. Such logical characterizations capture computational complexity without involving computation directly and suggest that logical expressibility of problems may determine their computational complexity. This is the direction taken by descriptive complexity theory [Imm89].

Such studies began with the work of Fagin who provided a logical characterization of NP [Fag74] as follows. Let L be an isomorphism closed class of finite structures over a vocabulary σ. L is NP-computable if and only if it is definable by an existential second-order sentence, i.e., if and only if there is a sentence $\phi(\mathbf{T})$ with predicate symbols amongst those in $\sigma \cup \mathbf{T}$ such that

$$\mathbf{A} \in L \Leftrightarrow \mathbf{A} \models (\exists \mathbf{T})\, \phi(\mathbf{T}),$$

where $\mathbf{A}$ is a finite structure over the vocabularly σ.

Since this caracterization of NP, researchers have provided logical characterization of many complexity classes; amongst these are characterizations of the classes P [Imm86, Var82], uniform versions of the circuit class AC_0 [BIS90], and PSPACE [Imm87b]. The important results in this field are surveyed in [Gur88, Imm87a, Imm89].

These are logical characterizations of language classes, or in other words the above work relates to defining a decision problem in a logical framework. With appropriate modifications, the logical definability framework can also be used to

* E-mail: saluja@tifrvax.bitnet.

† This author acknowledges the support of NSF Grant CCR-9108631. A major part of this work was carried out when this author was visiting the Tata Institute of Fundamental Research, Bombay. E-mail: thakur@cse.ucsc.edu.

0022-0000/95 $6.00

define classes of functions which map finite structures to natural numbers. For instance, Papadimitriou and Yannakakis [PY91] introduced the class MAX NP of optimization problems with finite structures as inputs. The full power of using first-order logic to study NP optimization problems was studied by Kolaitis and Thakur [KT93a, KT91]. They showed that an optimization problem $\mathcal{Q}$, with finite structures over a vocabulary σ as input, is a polynomially bounded NP maximization problem if and only if there is a first order formula $\phi(\mathbf{z}, \mathbf{T})$ with predicate symbols from the sequence $\mathbf{T} \cup \sigma$ and free first-order variables from a sequence $\mathbf{z}$, such that

$$\text{opt}_{\mathcal{Q}}(\mathbf{A}) = \max_{\mathbf{T}} |\{\mathbf{z} : \mathbf{A} \models \phi(\mathbf{z}, \mathbf{T})\}|.$$

Papadimitriou and Yannakakis [PY91] showed that every problem in the class MAX NP is approximable within a constant factor of the optimal by a polynomial time algorithm. This showed that there are interesting classes of functions defined using the framework of logical expressibility which have good computational behavior.

The purpose of this paper is to propose a logical characterization for the class #P of counting problems and to study the expressiveness and feasibility of syntax-restricted subclasses naturally obtained in this setting of logical definability. Valiant [Val79] defined #P to be the class of counting problems with an associated counting function on the input space. The counting function applied to an input is the number of accepting paths of an NP machine on that input. The class contains various natural and interesting counting problems such as #SAT (given a propositional formula, count the number of satisfying assignments), #HAMILTONIAN (given a graph, count the number of Hamiltonian cycles), and #EXT (given a poset, count the number of linear extensions). Since counting the number of solutions is at least as hard as checking if there is a solution, #P contains NP-hard problems. The hardness of problems in #P is further demonstrated by Toda's result [Tod89], which says that every language in the polynomial hierarchy [Sto76] can be recognized in deterministic polynomial time using at most one query to an oracle in #P. Seeing the apparent difficulty of computing #P functions, as well as their significance, researchers have used randomization to approximate some of these problems in polynomial time. In this direction, Karp *et al.* [KLM89] studied the notion of randomized approximability (see Definition 4.1) in polynomial time and showed that the #DNF problem has a fully polynomial time randomized approximation scheme (FPTRAS).

There have been other approaches, too, for approximating counting problems in #P which are of theoretical interest, e.g., enumerative counting [CH89], randomized approximation using NP oracles [VV86], deterministic polynomial time approximation using oracles in the polynomial hierarchy [Sto85].

Although #P problems are well studied using the model of Turing machines and polynomial time reducibilities, there are certain peculiarities which cannot be explained well in the setting of computational models. For example, both #DNF, #CNF (the value of the counting function is the number of satisfying assignments of a DNF, respectively CNF, propositional formula) are complete for #P under polynomial time Turing reductions; the former has a polynomial time randomized approximation algorithm [KLM89], whereas the latter does not have such an algorithm, unless RP = NP. The intuitive support for this difference, which one gets from the fact that the decision version of #CNF is NP-complete, while that of #DNF is in P, is not enough, because there are other counting problems (like counting the number of perfect matchings in a graph) whose decision versions are in P, but for which there is no known polynomial time randomized approximation algorithm.

Motivated by Fagin's characterization of NP, we examine the class of all problems $\mathcal{Q}$, with an associated counting function $f_{\mathcal{Q}}$, definable using first-order formulae $\phi(\mathbf{z}, \mathbf{T})\}|$,

$$f_{\mathcal{Q}}(\mathbf{A}) = |\{\langle \mathbf{T}, \mathbf{z} \rangle : \mathbf{A} \models \phi(\mathbf{z}, \mathbf{T})\}|,$$

where $\mathbf{A}$ is an ordered finite structure, $\mathbf{T}$ is a sequence of predicate variables, and $\mathbf{z}$ is a sequence of first-order variables. Throughout this paper we shall deal with ordered finite structures as inputs to the counting problems. These are finite structures along with $\leqslant$, a binary relation, which is always interpreted as a total order on the universe of the structure.

We show that on ordered finite structures the above framework captures exactly the class #P. In fact, $\#\Pi_2$, the syntactic subclass of the above class obtained by using only Π_2 formulae is #P. We study subclasses of $\#\Pi_2$ obtained by restricting the quantifier complexity of the first-order formulae involved. We obtain the classes $\#\Sigma_0$, $\#\Sigma_1$, $\#\Pi_1$, $\#\Sigma_2$ using Σ_0, Σ_1, Π_1, Σ_2 formulae, respectively.

In general, the set of properties definable by Σ_1 and Π_1 sentences are incomparable, but in this framework, it turns out that the class $\#\Sigma_1 \subseteq \#\Pi_1$. As a result we have the following hierarchy of classes:

$$\#\Sigma_0 = \#\Pi_0 \subseteq \#\Sigma_1 \subseteq \#\Pi_1 \subseteq \#\Sigma_2 \subseteq \#\Pi_2 = \#\text{P}.$$

Unlike the hierarchies of classes found in computational complexity theory, we prove nthat this is a true hierarchy, i.e., the five classes are distinct. These proofs are model theoretic in nature and do not take recourse to any complexity theoretic assumptions.

Next, we study the computational complexity of counting problems in these classes and get two positive results. We show that all the counting problems in $\#\Sigma_0$ are computable in deterministic polynomial time. These result seems levelwise optimal in the above hierarchy; i.e., it is not likely to be true for the levels above $\#\Sigma_0$ because the very next level $\#\Sigma_1$ has #P-complete problems. Next, we introduce *product reducibility*, which is an approximation preserving reducibility, and we show that all the problems in $\#\Sigma_1$ are product reducible to restricted versions of the #DNF problem. As a corollary, we show that every problem in $\#\Sigma_1$ has a fully polynomial time randomized approximation scheme (FPTRAS). In fact, the FPTRAS for a $\#\Sigma_1$ problem is very simple and does not require the full power of the method of Karp [KLM89]. Once again, this result is not likely to hold upwards in the hierarchy because the next level $\#\Pi_1$ contains the #3CNF problem which cannot have a polynomial time randomized approximation algorithm unless NP = RP. Nevertheless, we isolate a syntax-defined subclass $\#R\Sigma_2$ of $\#\Sigma_2$, which contains the #DNF problem and all the problems in it are *product reducible* to #DNF and, hence, have a FPTRAS.

The computational characteristics of the classes $\#\Sigma_0$ and $\#\Sigma_1$ are not likely to hold for the classes $\#\Pi_1$ and $\#\Sigma_2$ in the above logical hierachy. We consider a method of avoiding this situation by taking closure of these classes under product reductions. We show that the functions in #P form a 3-level hierarchy of closure classes in which the first two levels have counting functions with "feasible" computational properties, while the third is the whole of #P. By "feasible" we mean either polynomial time solvable or approximately by a polynomial time randomized approximation algorithm.

Although $\#\Sigma_0$ (and $\#\Sigma_1$) capture counting problems in #P which are computable in deterministic polynomial time (approximable by a randomized polynomial time algorithm, respectively), they cannot capture all such problems. There are polynomial time computable counting problems which are not in the class $\#\Sigma_1$. This prompts the following question. Given a counting problem specified in this framework, is the problem polynomial time computable, or does it have a polynomial time randomized approximation algorithm? We show that under reasonable complexity theoretic assumptions, such questions are undecidable. For example, assuming that $P \neq P^{\#P}$, it is an undecidable problem to determine for a given first-order formula ϕ, whether the associated counting function is polynomial time computable or not. Similar results follow for the existence of polynomial time randomized approximation algorithms.

The organization of the paper is as follows. In Section 2, we give the logical characterization of #P and in Section 3 we show that functions in #P form a logical hierarchy with five distinct levels. In Section 4, we show that the lower two levels of the hierarchy capture functions in #P which are polynomial time computable and approximate by polynomial time randomized algorithms, respectively, and also prove the undecidability results. In Section 5, we study the class $\#R\Sigma_2$ and the hierarchy of closure classes. In Section 6, we conclude with discussion of some open problems.

2. DESCRIPTIVE CHARACTERIZATION OF #P

In this section, we introduce a logic based framework for expressing counting problems and show that the class of functions definable in this framework is exactly the class #P. We formally define the notion of a counting problem as follows.

DEFINITION 2.1. A *counting problem* is a tuple $\mathcal{Q} = (\mathcal{I}_{\mathcal{Q}}, \mathcal{F}_{\mathcal{Q}}, f_{\mathcal{Q}})$ such that

- $\mathcal{I}_{\mathcal{Q}}$ is the set of input instances. $I_{\mathcal{Q}}$ is recognizable in polynomial time.
- $\mathcal{F}_{\mathcal{Q}}(I)$ is a set of feasible solutions for the input $I \in \mathcal{I}_{\mathcal{Q}}$.
- $f_{\mathcal{Q}}: \mathcal{I}_{\mathcal{Q}} \to \mathbf{N}$ is a counting function, corresponding to the problem $\mathcal{Q}$, and $f_{\mathcal{Q}}(I) = |\mathcal{F}_{\mathcal{Q}}(I)|$.

A counting Problem $\mathcal{Q}$ is in #P if there is an NP machine for which the number of accepting paths on input I is given by $f_{\mathcal{Q}}(I)$.

For this work, we shall assume that the instance space of a counting problem is a set of ordered finite structures over a certain vocabularly σ.

DEFINITION 2.2. A *vocabularly* (also known as a *finite similarity type*) $\sigma = \{\tilde{R}_1, ..., \tilde{R}_k\}$ is a finite set of predicate symbols. Each predicate symbol $\tilde{R}_i$ has a positive integer r_i as its designated arity. A *structure* $\mathbf{A} = (A, R_1, ..., R_k)$ over the vocabulary σ consists of a set A, called the universe of $\mathbf{A}$, and relations $R_1, ..., R_k$ of arities $r_1, ..., r_k$ on A, i.e., subsets of the Cartesian products $A^{r_1}, ..., A^{r_k}$, respectively, that are interpretations of the corresponding predicate symbols. A *finite* structure is a structure whose universe is a finite set. The *size* $|\mathbf{A}|$ of a finite structure $\mathbf{A}$ is the cardinality of its universe. An *ordered finite structure* is a finite structure with an extra relation $\leqslant$, which is always interpreted as a total-order on the elements of the universe of the structure.

Graphs are naturally represented as finite structures using a vocabulary with a single binary relation. Counting problems having inputs other than graphs can be represented as finite structures using an appropriate vocabulary.

We assume that the reader is familiar with the definitions of syntax and semantics of first-order logic over a vocabulary σ. We shall also not make a notational difference between the relations $\tilde{R}_i$ in the vocabulary and their interpretations R_i on the finite structure and denote both by R_i. It will be clear from the context which of the two we mean.

Henceforth we shall denote ordered finite structures by $\mathbf{A}$, their universe by A, and the total order by $\leqslant$. We shall always assume A to be the set $\{1, ..., n\}$, with the natural ordering on the elements, unless stated otherwise. We shall also use the notation $x \prec y$ to mean $(x \leqslant y) \wedge (x \neq y)$. We shall also use $\mathbf{T}$ to denote a finite sequence of predicate symbols, and $\mathbf{w}$, $\mathbf{x}$, $\mathbf{y}$, $\mathbf{z}$ to denote a finite sequence of first-order variables.

DEFINITION 2.3. Let σ be a vocabulary containing a relation symbol $\leqslant$. Let $\mathcal{Q}$ be a counting problem, with finite structures $\mathbf{A}$ over σ as instances. The relation $\leqslant$ is interpreted as a total order on the elements of the universe of $\mathbf{A}$. Let $\mathbf{T} = (T_1, ..., T_r)$, $r \geqslant 0$, be a sequence of predicate symbols and let $\mathbf{z} = (z_1, ..., z_m)$, $m \geqslant 0$, be a sequence of first-order variables such that at least one of the sequences is of non-zero length, i.e., $m + r > 0$. We say $\mathcal{Q}$ is in the class $\#\mathcal{F}\mathcal{O}$ if there is a first-order formula $\phi_{\mathcal{Q}}(\mathbf{z}, \mathbf{T})$, with predicate symbols from $\sigma \cup \mathbf{T}$ and free first-order variables from the sequence $\mathbf{z}$, such that

$$f_{\mathcal{Q}}(\mathbf{A}) = |\{\langle \mathbf{T}, \mathbf{z}\rangle : \mathbf{A} \models \phi_{\mathcal{Q}}(\mathbf{z}, \mathbf{T})\}|.$$

We define subclasses $\#\Pi_n$, $\#\Sigma_n$, $n \geqslant 0$ analogously using Π_n, Σ_n formulae, respectively, instead of arbitrary first-order formulae. Further, for a counting problem in $\#\mathcal{F}\mathcal{O}$, we refer to the following problem as the associated decision problem: Given an input structure $\mathbf{A}$, is there an interpretation of $\langle \mathbf{T}, \mathbf{z}\rangle$ on the structure $\mathbf{A}$ such that $\mathbf{A} \models \phi_{\mathcal{Q}}(\mathbf{T}, \mathbf{z})$?

Some examples of problems in these classes are as follows:

• #3CLIQUE. Given a graph $G = (V, E)$, the counting function is the number of triangles in the graph

$$f_{\#3\text{CLIQUE}}(G) = |\{\langle z_1, z_2, z_3\rangle : G \models z_1 \prec z_2 \wedge z_2 \prec z_3 \wedge E(z_1, z_2) \wedge E(z_2, z_3) \wedge E(z_1, z_3)\}|.$$

From the above formula it is clear that #3CLIQUE is in $\#\Sigma_0$.

• #3DNF. Given a Boolean formula in disjunctive normal form with at most three literals per disjunct, the counting function is the number of satisfying assignments. If there are less than three literals per disjunct, we just repeat some of them in that disjunct. An instance I of a 3DNF formula is encoded as a finite structure $\mathbf{A}(I)$ using a vocabulary $\{D_0, D_1, D_2, D_3\}$, where D_i, $i = 0, ..., 3$, are ternary relations. For $i = 0, ..., 3$, $D_i(x_1, x_2, x_3)$ if and only if $\{\neg x_1, ..., \neg x_i, x_{i+1}, ..., x_3\}$ appears as a disjunct in the 3DNF formula (cf. [KT93a]),

$$\begin{aligned} f_{\#3\text{DNF}}(\mathbf{A}(I)) = |\{\langle T\rangle : \mathbf{A}(I) \models (\exists x_1)(\exists x_2)(\exists x_3) \times (&D_0(x_1, x_2, x_3) \wedge T(x_1) \\ &\wedge T(x_2) \wedge T(x_3)) \\ &\vee (D_1(x_1, x_2, x_3) \\ &\wedge \neg T(x_1) \wedge T(x_2) \wedge T(x_3)) \\ &\vee (D_2(x_1, x_2, x_3) \\ &\wedge \neg T(x_1) \wedge \neg T(x_2) \wedge T(x_3)) \\ &\vee (D_3(x_1, x_2, x_3) \\ &\wedge \neg T(x_1) \wedge \neg T(x_2) \wedge \neg T(x_3))\}|. \end{aligned}$$

This shows that #3DNF is in the class $\#\Sigma_1$.

• #3CNF. Given a Boolean formula in conjunctive normal form with at most three literals per clause, the counting function is the number of satisfying assignments. An instance I of a 3CNF formula is encoded as a finite structure $\mathbf{A}(I)$ using a vocabulary $\{C_0, C_1, C_2, C_3\}$, where C_i, $i = 0, ..., 3$, are ternary relations. For $i = 0, ..., 3$, $C_i(x_1, x_2, x_3)$ if and only if $\{\neg x_1, ..., \neg x_i, x_{i+1}, ..., x_3\}$ appears as a clause in the 3CNF formula I (cf. [KT93a]),

$$\begin{aligned} f_{\#3\text{CNF}}(\mathbf{A}(I)) = |\{\langle T\rangle : \mathbf{A}(I) \models (\forall x_1)(\forall x_2)(\forall x_3) \times (&C_0(x_1, x_2, x_3) \\ &\wedge T(x_1) \vee T(x_2) \vee T(x_3)) \\ &\wedge (C_1(x_1, x_2, x_3) \\ &\rightarrow (\neg T(x_1) \vee T(x_2) \vee T(x_3))) \\ &\wedge (C_2(x_1, x_2, x_3) \\ &\rightarrow (\neg T(x_1) \vee \neg T(x_2) \vee T(x_3))) \\ &\wedge (C_3(x_1, x_2, x_3) \\ &\rightarrow (\neg T(x_1) \vee \neg T(x_2) \vee \neg T(x_3)))\}|. \end{aligned}$$

This shows that #3CNF is in the class $\#\Pi_1$.

• #DNF. The counting function for this problem is the number of satisfying assignments for a given DNF formula. We use the vocabulary $\{D, P, N\}$, with a unary relation D and two binary relations P, N, to encode a DNF formula I as a finite structure $\mathbf{A}(I)$. The structure $\mathbf{A}(I)$ has universe $A = D \cup V$, where V is the set of variables and D is the set of disjuncts of I. The predicate $P(d, v)$ ($N(d, v)$) expresses the fact that the disjunct d contains the variable v positively (negatively). Let T denote the set of variables assigned true value in a satisfying truth assignment to the instance I. Under this encoding,

$$\begin{aligned} f_{\#\text{DNF}}(\mathbf{A}(I)) = |\{S : \mathbf{A}(I) \models (\exists d)(\forall v)\, D(d) \wedge (&P(d, v) \\ &\rightarrow T(v)) \wedge (N(d, v) \rightarrow \neg T(v))\}|. \end{aligned}$$

Hence #DNF is in $\#\Sigma_2$.

• #CNF. Given a Boolean formula in conjunctive normal form, the counting function is the number of satisfying

assignments. We use the vocabulary $\{C, P, N\}$, with a unary relation C and two binary relations P, N, to encode a CNF formula I as a finite structure $\mathbf{A}(I)$. The structure $\mathbf{A}(I)$ has the universe $A = C \cup V$, where V is the set of variables and C is the set of clauses of I. The predicate $P(c, v)$ ($N(c, v)$) expresses the fact that clause c contains variable v positively (negatively). Let T denote the set of variables assigned true value in a satisfying truth assignment to the instance I. Under this encoding,

$$f_{\#\mathrm{CNF}}(\mathbf{A}(I)) = |\{T : \mathbf{A}(I) \models (\forall c)(\exists v)[\, C(c) \rightarrow ((P(c, v) \wedge T(v)) \vee (N(c, v) \wedge \neg T(v)))]\}|.$$

Hence #CNF is in the class $\#\Pi_2$.

- #HAMILTONIAN. Given a graph, the counting function is the number of Hamiltonian cycles in the graph. It follows as a consequence of the following theorem that #HAMILTONIAN is in $\#\Pi_2$. On can also demonstrate this fact directly, although it is not a trivial task to do so.

We now prove the following theorem which gives a descriptive characterization of the class #P.

THEOREM 1. *The class #P coincides with the class $\#\mathcal{F}\mathcal{O}$. In fact, the class #P is the class $\#\Pi_2$. Hence, $\#\mathrm{P} = \#\mathrm{P}_2 = \#\Pi_n = \#\Sigma_n$, $n > 2$.*

Proof. It is clear that if $\mathcal{Q}$ is a counting problem in $\#\mathcal{F}\mathcal{O}$, then $\mathcal{Q}$ is a problem in #P. For an input structure $\mathbf{A}$, the witnessing NP machine has to guess a tuple $\langle \mathbf{T}, \mathbf{z} \rangle$ and verify in polynomial time that $\mathbf{A} \models \phi_{\mathcal{Q}}(\mathbf{T}, \mathbf{z})$.

To prove the other direction, assume that $\mathcal{Q}$ is a problem in the class #P, the instances of $\mathcal{Q}$ being finite structures $\mathbf{A}$ over some vocabulary σ, which includes the relation symbol $\leqslant$. There is an NP machine for $\mathcal{Q}$ such tghat the number of accepting paths of the machine on input an encoding of $\mathbf{A}$ is given by $f_{\mathcal{Q}}(\mathbf{A})$. Hence, to check if $f_{\mathcal{Q}}(\mathbf{A})$ is nonzero is a problem in NP. By Fagin's characterization of NP [Fag74] in tems of definability in existential second-order logic, there is a first-order sentence $\phi(\mathbf{T})$, with relation symbols from σ and the sequence $\mathbf{T}$ of predicate symbols, such that $\mathbf{A}$ is in the NP language, i.e., $f_{\mathcal{Q}}(\mathbf{A}) \neq 0$, if and only if $\mathbf{A} \models (\exists \mathbf{T})\, \phi(\mathbf{T})$. It is straightforward to verify from the proof of Fagin's result that if $\leqslant$ is a built-in total order on the elements of the universe A, the formula ϕ can be chosen to be a Π_2 formula involving the binary relation symbol $\leqslant$. Further, the formula ϕ is such that, every accepting computation of the NP machine, on input an encoding of $\mathbf{A}$, corresponds to a unique value of the sequence $\mathbf{T}$ which satisfies $\phi(\mathbf{T})$. In other words, the number of accepting computations of the NP machine is exactly equal to $|\{\langle \mathbf{T} \rangle : \mathbf{A} \models \phi(\mathbf{T})\}|$. Hence $\mathcal{Q}$ is in the class $\#\Pi_2$. Therefore $\#\mathrm{P} = \#\mathcal{F}\mathcal{O} = \#\Pi_2$. ∎

It should be noted at this point that the built-in total order $\leqslant$ is crucial to the proof of the above theorem although it is not required in Fagin's characterization of the class NP [Fag74]. In the absence of a built-in order, one can quantigy out (as is done in Fagin's proof) using an existential quantifier, a binary relation $\leqslant$ and assert, as a subformula of ϕ, that $\leqslant$ represents a total order on the universe. However, any total order will suffice in satisfying the formula. There are $|\mathbf{A}|!$ possible binary relations which are total orders and hence the number of distinct assignment to $\mathbf{T}$ which satisfy $\phi(\mathbf{T})$ is $|\mathbf{A}|!$ times the number of accepting paths in the corresponding NP machine. This problem disappears when we use a unique built-in order $\leqslant$.

The above proof shows that counting the assignments to a sequence $\mathbf{T}$ of predicate variables gives the value of the function $f_{\mathcal{Q}}$, but we have considered a more generalized format in definition 2.3 which counts the assignments to $\langle \mathbf{T}, \mathbf{z} \rangle$, i.e., a sequence of second-order variables *and* a sequence of first-order variables. This is done because, as a number of examples show, there are counting functions in #P, which are more naturally expressible by counting the assignments of just first-order variables or a combination of first- and second-order variables.

3. LOGICAL HIERARCHY IN #P

Having shown that $\#\Pi_2$ captures the class #P, we now study the subclasses of $\#\Pi_2$ that are obtained by restricting the quantifier complexity of the first-order formulae. From the definition of the subclasses, $\#\Sigma_0 = \#\Pi_0$, $\#\Sigma_1$, $\#\Pi_1$, and $\#\Sigma_2$, one would expect that the containment between the classes is

$$\#\Sigma_0 = \#\Pi_0 \subseteq \begin{matrix} \#\Pi_1 \\ \\ \#\Sigma_1 \end{matrix} \subseteq \#\Sigma_2 \subseteq \#\Pi_2 = \#\mathrm{P}.$$

Contrary to such expectations, we show in this section, that the problems in $\#\Pi_2$ form a linear hierarchy with five distinct levels.

THEOREM 2. *The class $\#\Sigma_1$ is contained in the class $\#\Pi_1$. As a result,*

$$\#\Sigma_0 = \#\Pi_0 \subset \#\Sigma_1 \subset \#\Pi_1 \subset \#\Sigma_2 \subset \#\Pi_2 = \#\mathrm{P}.$$

Moreover, these containments are strict:

- #3DNF is in the class $\#\Sigma_1$ but not in the class $\#\Sigma_0$.
- #3CNF is in the class $\#\Pi_1$ but not in the class $\#\Sigma_1$.
- #DNF is in the class $\#\Sigma_2$ but not in the class $\#\Pi_1$.
- #HAMILTONIAN is in the class $\#\Pi_2$ but not in the class $\#\Sigma_2$.

Proof. We give this proof in five parts.

Part 1. We show here that $\#\Sigma_1$ is contained in $\#\Pi_1$. Let $\mathcal{Q}$ be a counting problem in $\#\Sigma_1$ with $f_{\mathcal{Q}}(\mathbf{A}) = |\{\langle \mathbf{T}, \mathbf{z}\rangle : \mathbf{A} \models (\exists \mathbf{x})\, \psi(\mathbf{x}, \mathbf{z}, \mathbf{T})\}|$, where $\psi(\mathbf{x}, \mathbf{z}, \mathbf{T})$ is a quantifier-free formula. Instead of counting the tuples $\langle \mathbf{T}, \mathbf{z}\rangle$ as above, we count the tuples $\langle \mathbf{T}, (\mathbf{z}, \mathbf{x}^*)\rangle$, where $\mathbf{x}^*$ is the lexicographically smallest $\mathbf{x}$, such that $\mathbf{A} \models \psi(\mathbf{x}, \mathbf{z}, \mathbf{T})$. It is clear that the latter count is also equal to $f_{\mathcal{Q}}(\mathbf{A})$. Let $\theta(\mathbf{x}^*, \mathbf{x}, \leqslant)$ be a quantifier-free formula that expresses the fact that $\mathbf{x}^*$ is lexicographically smaller than $\mathbf{x}$ under the built-in ordering $\leqslant$. Hence,

$$f_{\mathcal{Q}}(\mathbf{A}) = |\{\langle \mathbf{T}, (\mathbf{z}, \mathbf{x}^*)\rangle : \mathbf{A} \models \psi(\mathbf{z}, \mathbf{x}^*, \mathbf{T}) \wedge (\forall \mathbf{x})(\psi(\mathbf{x}, \mathbf{z}, \mathbf{T}) \rightarrow \theta(\mathbf{x}^*, \mathbf{x}, \leqslant)\}|.$$

As a result $\mathcal{Q} \in \#\Pi_1$, and, consequently $\#\Sigma_1 \subseteq \#\Pi_1$.

As $\#\Pi_2 = \#\mathrm{P}$, it follows that $\#\Sigma_2 \subseteq \Pi_2$. It is now clear that the subclasses of $\#\mathrm{P}$ form a hierarchy with five levels,

$$\#\Sigma_0 = \#\Pi_0 \subseteq \#\Sigma_1 \subseteq \#\Pi_1 \subseteq \#\Sigma_2 \subseteq \Pi_2 = \#\mathrm{P}.$$

In the following parts, we show that these classes are indeed distinct.

Part 2. We now show that #3DNF is not in the class $\#\Sigma_0$. We have already noted before that #3DNF is in the class $\#\Sigma_1$. Towards a contradiction, assume that #3DNF is in the class $\#\Sigma_0$. Therefore,

$$f_{\#\mathrm{3DNF}}(\mathbf{A}) = |\{\langle \mathbf{T}, \mathbf{z}\rangle : \mathbf{A} \models \psi(\mathbf{z}, \mathbf{T})\}|,$$

where $\psi(\mathbf{z}, \mathbf{T})$ is a Σ_0 (quantifier-free) formula and $\mathbf{A}$ is a structure representing 3DNF instances using an appropriate vocabulary. Let $\mathbf{T}$ consist of r relation symbols $T_1, \ldots, T_r$ and let $\mathbf{z}$ be of arity m.

We now prove that $r = 0$. Towards a contradiction, assume that $r > 0$. For all sufficiently large input structures $\mathbf{A}$ we show that $f_{\#\mathrm{3DNF}}(\mathbf{A})$ is even. Let $\mathbf{A}$, with universe A, be an input structure and let $\mathbf{z}^* \in A^m$. Since $\psi(\mathbf{z}, \mathbf{T})$ is a quantifier-free formula, to evaluate the satisfiability of $\psi(\mathbf{z}^*, \mathbf{T})$ we need only consider the truth values of T_i, $1 \leqslant i \leqslant r$, on a constant (independent of the size of $\mathbf{A}$) number of tuples of appropriate arity from A. Therefore if the structure $\mathbf{A}$ is sufficiently large, for every $\mathbf{z}^* \in A^m$, there are $t > 0$ tuples in A of appropriate arity such that the membership of these tuples in the relations T_i does not affect the satisfiability of $\psi(\mathbf{z}^*, \mathbf{T})$. Hence $|\{\mathbf{T} : \mathbf{A} \models \psi(\mathbf{z}^*, \mathbf{T})\}|$ is a multiple of 2^t for every $\mathbf{z}^* \in A^m$. Hence,

$$f_{\#\mathrm{3DNF}}(\mathbf{A}) = |\{\langle \mathbf{T}, \mathbf{z}\rangle : \mathbf{A} \models \psi(\mathbf{z}, \mathbf{T})\}| = \Sigma_{\mathbf{z}^* \in A^m} |\{\mathbf{T} : \mathbf{A} \models \psi(\mathbf{z}^*, \mathbf{T})\}|$$

is always even. In the proof of Theorem 3, the value of the counting function of a $\#\Sigma_0$ problem is estimated precisely.

But there are arbitrarily large 3DNF instances which have an odd number of satisfying assignments. For example, consider the 3DNF formula (actually a 1DNF formula), $\theta_n \equiv x_1 \vee x_2 \vee \cdots \vee x_n$ on n variables, $x_1, \ldots, x_n$. To note that θ_n has an odd number of satisfying assignments, we proceed as follows. Let $\mathcal{S}(n)$ be the number of satisfying assignments to the formula θ_n. It can be seen that $\mathcal{S}(1) = 1$ and $\mathcal{S}(k) = 2^{k-1} + \mathcal{S}(k-1)$, for $k > 1$. From this it follows, by induction, that $\mathcal{S}(n)$, the number of satisfying assignments to the formula θ_n, $n \geqslant 1$, is odd.

Hence, it follows that for the claimed $\#\Sigma_0$ function to represent #3DNF correctly it must be that $r = 0$. Hence, #3DNF can be expressed as

$$f_{\#\mathrm{3DNF}}(\mathbf{A}) = |\{\mathbf{z} : \mathbf{A} \models \psi(\mathbf{z})\}|.$$

Therefore, $f_{\#\mathrm{3DNF}}(\mathbf{A}) \leqslant |\mathbf{A}|^m$ for all structures $\mathbf{A}$. Since there are instances of 3DNF formulae whose number of satisfying assignments is at least an exponential in the size of the formula, we see that the $\#\Sigma_0$ function does not represent #3DNF correctly.

Part 3. We indicated before the #3CNF is in the class $\#\Pi_1$. We now show that #3CNF is not in the class $\#\Sigma_1$, thereby separating the classes $\#\Sigma_1$ and $\#\Pi_1$.

Let $\mathbf{A}$, with universe $A = \{x_1, \ldots, x_n\}$, $x_1 \prec \cdots \prec x_n$, be an ordered structure encoding a satisfiable instance of a 3CNF formula. Towards a contradiction, assume that #3CNF is in the class $\#\Sigma_1$. Hence $f_{\#\mathrm{3CNF}}(\mathbf{A}) = |\{\langle \mathbf{T}, \mathbf{z}\rangle : \mathbf{A} \models (\exists \mathbf{x})\, \psi(\mathbf{x}, \mathbf{z}, \mathbf{T})\}| \geqslant 1$, where $\psi(\mathbf{x}, \mathbf{z}, \mathbf{T})$ is a quantifier-free formula. Let $\langle \mathbf{T}^*, \mathbf{z}^*\rangle$ be such that $\mathbf{A} \models (\exists \mathbf{x})\, \psi(\mathbf{x}, \mathbf{z}^*, \mathbf{T}^*)$.

Let $\mathbf{B}$, with universe $B = \{x_1, \ldots, x_n, x_{n+1}\}$, $x_1 \prec \cdots \prec x_n \prec x_{n+1}$, be an unsatisfiable extension of the structure $\mathbf{A}$, with two new clauses, $\{x_{n+1}\}$ and $\{\neg x_{n+1}\}$. Since Σ_1 formulae are preserved under extensions, we have that $\mathbf{B} \models (\exists \mathbf{x})\, \psi(\mathbf{x}, \mathbf{z}^*, \mathbf{T}^*)$. Therefore, $f_{\#\mathrm{CNF}}(\mathbf{B}) \geqslant 1$, which is a contradiction. Therefore, #3CNF is not in the class $\#\Sigma_1$.

Part 4. We now prove that the #DNF problem is not in the class $\#\Pi_1$. We mentioned before that this problem is in the class $\#\Sigma_2$. Towards a contradiction, assume that #DNF is in the class $\#\Pi_1$. Therefore, there is a quantifier-free formula ψ such that

$$f_{\#\mathrm{DNF}}(\mathbf{A}) = |\{\langle \mathbf{T}, \mathbf{z}\rangle : \mathbf{A} \models (\forall \mathbf{x})\, \psi(\mathbf{x}, \mathbf{z}, \mathbf{T})\}|,$$

where $\mathbf{A}$ is an input structure representing a DNF instance using the vocabulary $\{D, P, N\}$. Let us denote the arity of

$\mathbf{z}$ by m and fix a positive integer $p > 2m+1$. Consider the DNF formula

$$F \equiv (w_1 \wedge w_2 \wedge \cdots w_p \wedge \neg w_1) \vee (w_1 \wedge w_2 \wedge \cdots w_p \wedge \neg w_2) \vee \cdots (w_1 \wedge w_2 \wedge \cdots w_p \wedge \neg w_p)$$

represented by the finite structure $\mathbf{A} = (U, D, P, N)$, where

$$U = \{d_1, ..., d_p, w_1, ..., w_p\},$$
$$D = \{d_1, ..., d_p\},$$
$$P = \{\langle d_1, w_1\rangle, ..., \langle d_1, w_p\rangle, \langle d_2, w_1\rangle, ..., \langle d_2, w_p\rangle, ..., \langle d_p, w_1\rangle, ..., \langle d_p, w_p\rangle\},$$
$$N = \{\langle d_1, w_1\rangle, ..., \langle d_p, w_p\rangle\}.$$

Let $\mathbf{A}_i$, $i = 1, ..., p$, be the substructure of $\mathbf{A}$ induced by $U - \{w_i\}$. Note that $\mathbf{A}_i$ represents the formula

$$F_i \equiv (w_1 \wedge \cdots w_{i-1} \wedge w_{i+1} \wedge \cdots \wedge w_p \wedge \neg w_1) \vee \cdots (w_1 \wedge \cdots w_{i-1} \wedge w_{i+1} \wedge \cdots w_p \wedge \neg w_{i-1}) \vee (w_1 \wedge \cdots w_{i-1} \wedge w_{i+1} \wedge \cdots w_p) \vee (w_1 \wedge \cdots w_{i-1} \wedge w_{i+1} \wedge \cdots w_p \wedge \neg w_{i+1}) \vee \cdots (w_1 \wedge \cdots w_{i-1} \wedge w_{i+1} \wedge \cdots w_p \wedge \neg w_p).$$

The dnf formula F_i is satisfiable by setting the variables $w_1, ..., w_{i-1}, w_{i-1}, w_{i+1}, ..., w_p$ to true. Therefore, $f_{\#\mathrm{DNF}}(\mathbf{A}_i) = 1$, for $i = 1, ..., p$. Let $\langle \mathbf{T}_i^*, \mathbf{z}_i^*\rangle$ be the tuple on $\mathbf{A}_i$ such that $\mathbf{A}_i \models (\forall \mathbf{x})\, \psi(\mathbf{x}, \mathbf{z}_i^*, \mathbf{T}_i^*)$, $i = 1, ..., p$. It can be seen that d_i must be present as a component of $\mathbf{z}_i^*$. If such is not the case, then in the substructure $\mathbf{B}_i$ of $\mathbf{A}_i$, obtained by deleting d_i, we would have

$$\mathbf{B}_i \models (\forall \mathbf{x})\, \psi(\mathbf{x}, \mathbf{z}_i^*, \mathbf{T}_i'),$$

where $\mathbf{T}_i'$ is sequence of appropriately induced subrelations of the relations in $\mathbf{T}^*$. So we would have $f_{\#\mathrm{DNF}}(\mathbf{B}_i) \geq 1$, which is untrue, as B_i represents the formula A_i without the disjunct $(w_1 \wedge \cdots w_{i-1} \wedge w_{i+1} \wedge \cdots w_p)$.

Since the arity of $\mathbf{z}_i^*$ is m and $p > 2m+1$, it follows that $\binom{p}{2} > mp$. It can be seen that there exist $u, v, u \neq v$, $1 \leq u, v \leq p$, such that $\mathbf{z}_u^*$ does not have d_v as a component and $\mathbf{z}_v^*$ does not have d_u as a component.

Let $\mathbf{A}_{u,v}$ be the substructure of $\mathbf{A}_u$ obtained by deleting $\{w_v\}$ from $\mathbf{A}_u$ and let $\mathbf{A}_{v,u}$ be the subformula of $\mathbf{A}_v$ obtained by deleting $\{w_u\}$ from $\mathbf{A}_v$. It is clear that $\mathbf{A}_{u,v}$ and $\mathbf{A}_{v,u}$ represent the same formula. Further, the formula has exactly one satisfying assignment.

Let $\langle \mathbf{T}_{u,v}^*, \mathbf{z}_{u,v}^*\rangle$ be the substructure of $\langle \mathbf{T}_u^*, \mathbf{z}_u^*\rangle$ induced by $\mathbf{A}_{u,v}$, and let $\langle \mathbf{T}_{v,u}^*, \mathbf{z}_{v,u}^*\rangle$ be the substructure of $\langle \mathbf{T}_v^*, \mathbf{z}_v^*\rangle$ induced by $\mathbf{A}_{v,u}$. Since $\mathbf{z}_u^*$ has d_u, but not d_v, as a component and $\mathbf{z}_v^*$ has d_v, but not d_u, as a component, $\langle \mathbf{T}_{u,v}^*, z_{u,v}^*\rangle$ and $\langle \mathbf{T}_{v,u}^*, \mathbf{z}_{v,u}^*\rangle$ are distinct. As universal formulae are preserved under substructures, we have that

$$\mathbf{A}_{u,v} \models (\forall \mathbf{x})\, \psi(\mathbf{x}, \mathbf{z}_{u,v}^*, \mathbf{T}_{u,v}^*),$$
$$\mathbf{A}_{v,u} \models (\forall \mathbf{x})\, \psi(\mathbf{x}, \mathbf{z}_{v,u}^*, \mathbf{T}_{v,u}^*).$$

As a result, $f_{\#\mathrm{DNF}}(\mathbf{A}_{u,v}) \geq 2$, which is a contradiction. Therefore #DNF is not in the class $\#\Pi_1$.

Part 5. We now show that #HAMILTONIAN is not in the class $\#\Sigma_2$. Towards a contradiction, let us assume that #HAMILTONIAN is in the class $\#\Sigma_2$; i.e., there is a quantifier-free formula $\psi(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{T})$ such that

$$f_{\#\mathrm{HAMILTONIAN}}(G) = |\{\langle \mathbf{T}, \mathbf{z}\rangle : G \models (\exists \mathbf{x})(\forall \mathbf{y})\, \psi(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{T})\}|.$$

Let m be the arity of $\mathbf{z}$ and let t be the arity of $\mathbf{x}$.

Now consider a graph G which is a cycle of $n = m + t + 1$ vertices. Since $f_{\#\mathrm{HAMILTONIAN}}(G) = 1$, there are tuples $\langle \mathbf{T}^*, \mathbf{z}^*\rangle$ and $\mathbf{x}^*$ such that $G \models (\forall \mathbf{y})\, \psi(\mathbf{x}^*, \mathbf{z}^*, \mathbf{T}^*)$. It is now clear that there is one vertex in G, say a, such that a does not appear as a component in either $\mathbf{x}^*$ or $\mathbf{z}^*$. Let G_1 be the subgraph of G obtained by deleting a and the two edges incident on a. Note that G_1 is not Hamiltonian, but that $G_1 \models (\forall \mathbf{y})\, \psi(\mathbf{x}^*, \mathbf{y}, \mathbf{z}^*, \mathbf{T}_1^*)$, where $\mathbf{T}_1^*$ is the subset of $\mathbf{T}^*$ obtained by deleting all occurrences of a from $\mathbf{T}^*$. Therefore,

$$f_{\#\mathrm{HAMILTIONIAN}}(G_1) = |\{\langle \mathbf{T}, \mathbf{z}\rangle : G_1 \models (\exists \mathbf{x})(\forall \mathbf{y})(\psi(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{T})\}| \geq 1,$$

which is a contradiction. Therefore #HAMILTONIAN is not in the class $\#\Sigma_2$. ∎

Remark 1. Using model theoretic arguments similar to those used in the above proof, the counting problems #CNF and #PM (counting the number of perfect matchings in a bipartite graph) can be shown to lie in $\#\Pi_2 - \#\Sigma_2$.

4. FEASIBILITY OF LOGICALLY DEFINED SUBCLASSES

One of our objectives in using logical definability to study counting problems is to obtain syntactically defined subclasses of #P which contain *computationally feasible* problems. By computationally feasible we mean in this paper, either polynomial time computable, or approximable within a constant factor by a polynomial time randomized algorithm. In this section, we first show that every problem

in the class $\#\Sigma_0$ is polynomial time computable and that every problem in the class $\#\Sigma_1$ is approximable in polynomial time by a randomized algorithm. The above classes give only sufficient conditions for polynomial time computability and randomized approximability of counting functions. We show that under reasonable complexity theoretic assumptions, it is an undecidable problem to tell if a given formula defines a polynomial time counting problem or a problem that is approximable by a polynomial time randomized algorithm. It should be pointed out that these results do not contradict the fact that there is an effective syntax for the class of counting problems that are computable in polynomial time and they do not rule out the possibility of an effective syntax for the class of all approximable counting functions. Recently, Kolaitis and Thakur have provided a descriptive characterization of the class of all polynomial time computable functions [KT93b] in terms of stage functions of positive first-order formulae.

THEOREM 3. *Every counting problem in $\#\Sigma_0$ is computable in deterministic polynomial time.*

Proof. Let $\mathcal{Q}$ be a counting problem in $\#\Sigma_0$ which is defined on ordered finite structures over a vocabulary σ. Then there is a quantifier-free formula $\psi_{\mathcal{Q}}$ such that

$$f_{\mathcal{Q}}(\mathbf{A}) = |\{\langle \mathbf{T}, \mathbf{z}\rangle : \mathbf{A} \models \psi_{\mathcal{Q}}(\mathbf{z}, \mathbf{T})\}|,$$

where $\mathbf{A}$ is a finite ordered structure over the vocabulary σ, $\mathbf{T}$ is a sequence $(T_1, T_2, ..., T_r)$ of second-order predicate variables of arities $a_1, a_2, ..., a_r$, respectively, and $\mathbf{z}$ is an m-tuple $(z_1, ..., z_m)$.

To compute $f_{\mathcal{Q}}(\mathbf{A})$, we count, for each $\mathbf{z}^* \in \mathbf{A}^m$, the number of assignments to $\mathbf{T}$ so that $\mathbf{A} \models \psi_{\mathcal{Q}}(z^*, \mathbf{T})$. The number of such $\mathbf{z}^*$ is $|\mathbf{A}|^m$. To show that $\mathcal{Q}$ is polynomial time computable, it suffices to show that for every $\mathbf{z}^* \in \mathbf{A}^m$, we can compute $f'_{\mathcal{Q}}(\mathbf{z}^*) = |\{\mathbf{T} : \psi(\mathbf{z}^*, \mathbf{T})\}|$ in polynomial time. Then, $f_{\mathcal{Q}}(\mathbf{A}) = \sum_{\mathbf{z}^* \in \mathbf{A}^m} f'_{\mathcal{Q}}(\mathbf{z}^*)$.

For every $\mathbf{z}^* {}'A^m$, consider the formula $\psi_{\mathcal{Q}}(\mathbf{z}^*, \mathbf{T})$. The formula $\psi_{\mathcal{Q}}(\mathbf{z}^*, \mathbf{T})$ can also be viewed as a propositional formula with variables of the form $T_i(\mathbf{y}_i)$, where $\mathbf{y}_i$ are tuples of arity a_i, $i = 1, ..., r$. The total number of such variables is $\sum_{i=1}^{r} |\mathbf{A}|^{a_i}$. Let $c(\mathbf{z}^*)$ denote the number of such variables such that either the variable or its negated literal appear in the formula $\psi_{\mathcal{Q}}(\mathbf{z}^*, \mathbf{T})$. In other words, c($\mathbf{z}^*$) is the total number of a_i-tuples $\mathbf{y}_i$, for $i = 1, ..., r$, such that either $T_i(\mathbf{y}_i)$ or $\neg T_i(\mathbf{y}_i)$ appear in the formula $\psi_{\mathcal{Q}}(\mathbf{z}^*, \mathbf{T})$. Note that $c(\mathbf{z}^*)$ is bounded above by a constant, i.e., it does not depend on the size of the structure $\mathbf{A}$. Hence, we can find all the truth assignments to the variables that occur in this propositional formula in constant time. Let the number of such satisfying assignments be $s(\mathbf{z}^*)$. The number of propositional variables that do not appear in this propositional formula is $(\sum_{i=1}^{r} |\mathbf{A}|^{a_i}) - c(\mathbf{z}^*)$. It is easily seen that

$$f'_{\mathcal{Q}}(\mathbf{z}^*) = s(\mathbf{z}^*)\, 2^{[(\sum_{i=1}^{r} |\mathbf{A}|^{a_i}) - c(\mathbf{z}^*)]},$$

which can be computed in polynomial time. ∎

Remark 2. If r, the number of peredicate symbols in the sequence $\mathbf{T}$, is greater than zero then one can choose sufficiently large structures $\mathbf{A}$ such that $|\mathbf{A}|^{a_i} - c(\mathbf{z}^*) > 0$, $i = 1, ..., r$, for all $\mathbf{z}^* \in A^m$. Then, the value of $f'_{\mathcal{Q}}(\mathbf{z}^*)$ is always a multiple of a power of 2, including zero, and hence $f_{\mathcal{Q}}(\mathbf{A})$ is always even. This fact was crucially used in Part 2 of the proof of Theorem 2.

Remark 3. It is unlikely that every problem in the next higher class in the hierarchy, i.e., $\#\Sigma_1$ has only polynomial time computable problems, because $\#\Sigma_1$ has $\#$P-complete problems, e.g., $\#$3DNF.

We now study the computational properties of the class $\#\Sigma_1$. We show that every problem in $\#\Sigma_1$ has a polynomial time randomized approximation algorithm. The following definition makes precise what we mean by polynomial time randomized approximation.

DEFINITION 4.1. A counting problem $\mathcal{Q}$ is said to have a *polynomial time randomized* (ε, δ) *approximation algorithm*, if there is a randomized algorithm M and there are constants $\varepsilon, \delta > 0$ such that for all inputs $I \in \mathcal{I}_{\mathcal{Q}}$,

1. $\Pr[|(M(I) - f_{\mathcal{Q}}(I)/f_{\mathcal{Q}}(I)| > \varepsilon] < \delta$. We assume, without loss of generality, that $f_{\mathcal{Q}}(I) \neq 0$;

2. the running time of M on input I is bounded by a polynomial in $|I|$.

If M satisfies the above two conditions for every $\varepsilon, \delta > 0$, then $\mathcal{Q}$ is said to have a *polynomial time randomized approximation scheme*. Further, if the running time of M is bounded by a plynomial in $1/\varepsilon$, $1/\delta$, then $\mathcal{Q}$ is said to have a *fully polynomial time randomized approximation scheme* (FPTRAS).

In the spirit of this paper, we shall use finite structures $\mathbf{A}$ as inputs to the algorithms, and replace I by $\mathbf{A}$ in the above definition.

We show the existence of a FPTRAS for every problem in $\#\Sigma_1$ in two steps:

(1) Every problem in $\#\Sigma_1$ is reducible to a restricted version of $\#$DNF problem, under a reducibility which preserves approximability.

(2) The restricted version of the $\#$DNF has a FPTRAS.

Before we achieve step 1 above, we define the appropriate notion of reduction and briefly comment on its properties.

DEFINITION 4.2. Given counting problems $\mathcal{Q}$, $\mathcal{R}$, we say $\mathcal{Q}$ is *polynomial time product reducible* to $\mathcal{R}$ (written as $\mathcal{Q} \leqslant_{pr} \mathcal{R}$) if there are polynomial time computable functions g, h,

$$g : I_{\mathcal{Q}} \to I_{\mathcal{R}}, \quad h : \mathbf{N} \to \mathbf{N},$$

such that for every finite structure $\mathbf{A}$, with universe A, which is an input to $\mathcal{Q}$, the value of the counting function is given by $f_{\mathcal{Q}}(\mathbf{A}) = f_{\mathcal{R}}(g(\mathbf{A})) \times h(|\mathbf{A}|)$. If h is the constant 1 function, the reduction is said to be *parsimonious*.

Throughout this paper, we will use *product reducible* to mean polynomial time product reducible.

PROPOSITION 1. *Given counting problems $\mathcal{Q}$, $\mathcal{R}$, if $\mathcal{Q} \leqslant_{pr} \mathcal{R}$ and $\mathcal{R}$ is computable in polynomial time, then $\mathcal{Q}$ is computable in polynomial time.*

PROPOSITION 2. *Given functions $\mathcal{Q}$, $\mathcal{R}$, if $\mathcal{Q} \leqslant_{pr} \mathcal{R}$ and $\mathcal{R}$ has a polynomial time randomized (ε, δ) approximation algorithm, then $\mathcal{Q}$ also has a polynomial time randomized (ε, δ) approximation algorithm.*

DEFINITION 4.3. For any positive constant k, the counting problem $\#k \cdot \text{logDNF}$ ($\#k \cdot \text{logCNF}$) is the problem of counting the number of satisfying assignments to a propositional formula in disjunctive normal form (respectively, conjunctive normal form) in which the number of literals in each disjunct (respectively, conjunct) is at most $k \log(n)$, where n is the number of propositional variables in the formula.

LEMMA 1. *For every counting problem $\mathcal{Q} \in \#\Sigma_1$, there is positive constant k so that $\mathcal{Q} \leqslant_{pr} \#k \cdot \text{logDNF}$.*

Proof. Let $\mathcal{Q}$ be a problem in $\#\Sigma_1$ such that $f_{\mathcal{Q}}(\mathbf{A}) = |\{\langle \mathbf{T}, \mathbf{z} \rangle : \mathbf{A} \models (\exists \mathbf{y})\, \psi(\mathbf{y}, \mathbf{z}, \mathbf{T})\}|$, where $\psi(\mathbf{y}, \mathbf{z}, \mathbf{T})$ is a quantifier-free formula in DNF form, with at most t literals per disjunct, and $\mathbf{T}$ is a sequence $(T_1, T_2, \ldots, T_r)$ of second-order predicate variables whose arities are $a_1, a_2, \ldots, a_r$, respectively. Let p denote the arity of $\mathbf{y}$ and let m denote the arity of $\mathbf{z}$. Let $\{\mathbf{y}_1, \ldots, \mathbf{y}_{|\mathbf{A}|^p}\}$ be the set A^p (note that $|\mathbf{A}|$ is the size of the universe A of the structure $\mathbf{A}$) and let $\{\mathbf{z}_0, \ldots, \mathbf{z}_{|\mathbf{A}|^m - 1}\}$ be the set A^m.

For every $\mathbf{z}_i \in A^m$, we write the formula $(\exists \mathbf{y})\, \psi(\mathbf{y}, \mathbf{z}_i, \mathbf{T})$ as a disjunction $\bigvee_{j=1}^{|\mathbf{A}|^p} \psi(\mathbf{y}_j, \mathbf{z}_i, \mathbf{T})$. Let $\psi'(\mathbf{z}_i, \mathbf{T})$ denote the formula obtained from $\bigvee_{j=1}^{|\mathbf{A}|^p} \psi(\mathbf{y}_j, \mathbf{z}_i, \mathbf{T})$ by replacing every subformula that is satisfied by $\mathbf{A}$ by the value TRUE and every subformula that is not satisfied by $\mathbf{A}$ by FALSE. It is clear that $\psi'(\mathbf{z}_i, \mathbf{T})$ is a propositional formula in DNF with variables of the form $T_i(\mathbf{w}_i)$, with $\mathbf{w}_i \in A^{a_i}$ and $1 \leqslant i \leqslant r$.

We define new propositional variables $x_1, x_2, \ldots, x_l$, where $2^{l-1} < |\mathbf{A}|^m \leqslant 2^l$. For every $\mathbf{s} \in \{0, 1\}^l$, let $x(\mathbf{s})$ represent the conjunction of these l variables so that for $1 \leqslant i \leqslant l$, x_i appears complemented in $x(\mathbf{s})$ if and only if the ith component of $\mathbf{s}$ is 0. We shall interpret $\mathbf{s}$ as the binary representation of an integer between 0 and $2^l - 1$.

Consider now the propositional formula $\theta_{\mathbf{A}}$ defined as

$$\theta_{\mathbf{A}} \stackrel{\text{def}}{\equiv} [\psi'(\mathbf{z}_0, \mathbf{T}) \wedge x(0)] \vee [\psi'(\mathbf{z}_1, \mathbf{T}) \wedge x(1)] \vee \cdots \vee [\psi'(\mathbf{z}_{|\mathbf{A}|^m - 1}, \mathbf{T}) \wedge x(|\mathbf{A}|^m - 1)],$$

with one term for each $\mathbf{z}_i \in A^m$. This is a DNF formula with variables $x_1, x_2, \ldots, x_l$ and $T_i(\mathbf{w}_i)$, with $\mathbf{w}_i \in \mathbf{A}^{a_i}$. Let $c(\mathbf{A})$ be the number of variables among these that do not appear in $\theta_{\mathbf{A}}$. It is straightforward to check that

(i) $\theta_{\mathbf{A}}$ is a propositional formula in disjunctive normal form with at most $t + l$ literals per disjunct. Since $l = O(\log(n))$, $\theta_{\mathbf{A}}$ is a $k \cdot \text{logDNF}$ formula for suitable k depending on the size of $\psi(\mathbf{y}, \mathbf{z}, \mathbf{T})$.

(ii) $f(\mathbf{A}) = 2^{c(\mathbf{A})} \times$ (the number of satisfying assignments of $\theta_{\mathbf{A}}$).

Finally, note that we can construct $\theta_{\mathbf{A}}$ in polynomial time. ∎

Remark 4. By a similar proof, it can also be shown that every counting problem in $\#\Pi_1$ is product reducible to $\#k \cdot \text{logCNF}$.

Remark 5. Let $\mathcal{Q}$ be a counting problem in $\#\Sigma_1$. The decision version of the counting problem $\mathcal{Q}$ is: Given a finite structure $\mathbf{A}$, is $f_{\mathcal{Q}}(\mathbf{A}) \neq 0$? This has a YES answer if and only if $\mathbf{A} \models (\exists \mathbf{T})(\exists \mathbf{z})\, \phi_{\mathcal{Q}}(\mathbf{T}, \mathbf{z})$, where $\phi_{\mathcal{Q}}$ is an existential formula. Corresponding to the formula $(\exists \mathbf{T})(\exists \mathbf{z})\, \phi_{\mathcal{Q}}(\mathbf{T}, \mathbf{z})$, there exists a first-order existential formula ϕ' such that $\mathbf{A} \models (\exists \mathbf{T})(\exists \mathbf{z})\, \phi_{\mathcal{Q}}(\mathbf{T}, \mathbf{z})$ if and only if $\mathbf{A} \models \phi'$. We know from definability theory that the problem of model checking for a first order formula is in the uniform version of the class AC^0 [BIS90, Imm87a]. Hence it follows that the complexity of the decision version of every $\#\Sigma_1$ problem is very low, in fact in AC^0.

LEMMA 2. *For every k, there is a fully polynomial time randomized approximation scheme for the $\#k \cdot \text{logDNF}$ problem.*

The above lemma is a special case of the result of Karp *et al.* [KLM89], who give a FPTRAS for the general #DNF problem. However, as our proof below shows, there is a simple FPTRAS which works for the $\#k \cdot \text{logDNF}$ problem (for every k) and does not need the full power of the method of Karp *et al.* Before giving the proof, we recall a result on a Chernoff-type bound for random variables. A good reference for Chernoff bounds is [HR90].

LEMMA 3. *Let $x_1, x_2, \ldots, x_t$ be t independent and identically distributed $\{0, 1\}$ valued random variables each having expected value $p < 0.5$. Then for every $\varepsilon > 0$, $\Pr[|(x_1 + x_2 + \cdots + x_t)/t - p| > \varepsilon p] < 2 \exp(-2\varepsilon^2 pt/9(1-p))$.*

The proof of this lemma follows from Theorem 2, page 41 of Rényi [R70] and was used by Karp *et al.* [KLM89] in their analysis of a FPTRAS for the #DNF problem.

Proof of Lemma 2. Let d be a DNF formula so that there are at most $k \log(n)$ literals per disjunct, where n is the number of propositional variables occurring in the formula. If d is satisfiable, (which can be checked in polynomial time), then consider any disjunct which is satisfiable. Observe that this disjunct is satisfiable for at least $1/2^{k \log(n)}$ fraction of all the satisfying assignments. Therefore if an assignment is chosen randomly from the uniform distribution over all the assignments, the probability that it is a satisfying assignment of d is at least $1/n^k$.

Define a random variable x as follows. Choose a random vector v from the uniform distribution on $\{0, 1\}^n$. If v represents a satisfying assignment of d, then set x to 1 else set x to 0. Therefore, the expected value p of the random variable x is $\geq 1/n^k$. Consider the following procedure for estimating the number of satisfying assignments of d.

$X = 0$.
For $i = 1$ to t
begin
Pick a random vector from $\{0, 1\}^n$
If it represents a satisfying assignment of d, set $X := X + 1$
end
Output $2^n \times (X/t)$.

Using Lemma 3, $\Pr[\,|X/t - p| > \varepsilon p\,] < 2 \exp(-2\varepsilon^2 pt/9(1-p))$. If we choose the number of trials t to be $(9(1-p)/2p\varepsilon^2) \log(2/\delta)$, then the above probability is less than δ. Note that t is bounded by a polynomial in n, $1/\varepsilon$, $\log(1/\delta)$. ∎

The next Theorem now follows from Lemmas 1 and 2.

THEOREM 4. *Every counting function in $\#\Sigma_1$ has a FPTRAS.*

Remark 6. It is unlikely that every problem in the next higher class in the hierarchy, viz., $\#\Pi_1$, has a polynomial time randomized approximation algorithm. Such a result would imply that NP = RP, since #3CNF lies in the class $\#\Pi_1$ and its decision version 3SAT is NP-complete.

The class $\#\Sigma_1$ has a lot of interesting problems like, #MONOCHROMATIC-kCLIQUE-PARTITIONS, #NON-VERTEX-COVERS, #NON-CLIQUES. It follows from Theorem 4 that all these problems have a FPTRAS.

We have seen that the syntax of a first-order formula used in defining the #P problems has some impact on the computational complexity of the corresponding counting function. A natural question to ask in this context is: Given an arbitrary first order formula $\phi(\mathbf{T}, \mathbf{z})$, is the counting problem defined using this formula polynomial time computable, or is the problem approximable by a polynomial time (ε, δ) randomized algorithm? We show below that these are undecidable problems.

Before we state and prove the theorem, we need the following definitions of relativized formulae from logic.

DEFINITION 4.4. If φ is a first-order formula and R is a unary predicate, then the *relativized formula* φ^R is obtained from φ by replacing every universal subformula $(\forall x)\,\varphi'(x)$ of φ by $(\forall x)(R(x) \to \varphi'(x))$ and by replacing every existential subformula $(\exists x)\,\varphi'(x)$ of φ by $(\exists x)(R(x) \wedge \varphi'(x))$.

Let σ be a vocabulary and let $\phi(\mathbf{z}, \mathbf{T})$ be a first-order formula with predicate symbols from $\sigma \cup \mathbf{T}$. We say a counting problem $\mathcal{Q}_\phi$ with instances finite structures over σ *is defined by the formula* $\phi(\mathbf{z}, \mathbf{T})$, if the counting function $f_{\mathcal{Q}}$ is defined using the formula ϕ as

$$f_{\mathcal{Q}}(\mathbf{A}) = |\{\langle \mathbf{T}, \mathbf{z} \rangle : \mathbf{A} \models \phi(\mathbf{z}, \mathbf{T})\}|.$$

We also write f_ϕ for this function.

THEOREM 5. *Let σ be a vocabulary with a unary predicate $\{C\}$ and three binary predicate symbols $\{E, P, N\}$.*

(a) *Assuming* NP ≠ RP, *the following is an undecidable problem: Given a first-order formula $\phi(\mathbf{z}, \mathbf{T})$ over $\sigma \cup \mathbf{T}$, does the counting problem $\mathcal{Q}_\phi$ have an polynomial time (ε, δ) randomized approximation algorithm, for some constants $\varepsilon, \delta > 0$?*

(b) *Similarly, assuming* $\mathrm{P} \neq \mathrm{P}^{\#\mathrm{P}}$, *the following is an undecidable problem: Given a first-order formula $\phi(\mathbf{z}, \mathbf{T})$ over $\sigma \cup \mathbf{T}$, is the counting problem $\mathcal{Q}_\phi$ polynomial time computable?*

Proof. (a) The proof is similar to that given in [KT91] to show an analogous result for NP optimization problems. We will use Trakhtenbrot's theorem which asserts that for any vocabulary τ with at least one binary symbol, the set of first-order sentences true on all finite structures over τ is not recursive. Assuming NP ≠ RP, we will give a reduction from the above problem to the problem of deciding "approximability," thereby showing the undecidability of the latter.

Consider the #CNF problem and assume that the instances of theCNF formula are encoded as finite structures $\mathbf{A}$ over vocabulary $\{C, P, N\}$ (see Section 2), with universe A. Let

$$f_{\#\mathrm{CNF}}(\mathbf{A}) = |\{\langle T \rangle : \mathbf{A} \models \phi(T)\}|.$$

Given an arbitrary first-order formula ψ over vocabulary $\{E\}$, define the first-order $F_\psi(T)$ as: $F_\psi(T) \equiv^{\mathrm{def}} \neg\psi^V \wedge \phi^{V'}(T) \wedge (T \subseteq V')$, where V' is the complement $\neg V$ of V. Let $\mathcal{Q}_{F_\psi}$ be the problem defined by F_ψ, which has as inputs,

finite structures of the form $\mathbf{B} = (B, V, C, E, P, N)$, with the universe B. Consider f_{F_ψ}, the counting function of $\mathcal{Q}_{F_\psi}$ defined as

$$f_{F_\psi}(\mathbf{B}) = |\{\langle T\rangle : \mathbf{B} \models F_\psi(T)\}|.$$

Observe that, if ψ is true over all finite structures, then $F_\psi(T)$ is false over all finite structures and the counting function f_{F_ψ} has answer 0 on all structures $\mathbf{B}$. Hence the problem $\mathcal{Q}_{F_\psi}$ is trivially approximable.

On the other hand, if ψ is false on some finite structure, say (V, E), with universe V, then there is a polynomial time parsimonious reduction from #CNF to the problem $\mathcal{Q}_{F_\psi}$. Therefore assuming that NP $\neq$ RP, #CNF, and hence $\mathcal{Q}_{F_\psi}$, is not approximable by any randomized ε, δ approximation algorithm. The parsimonious reduction from #CNF to $\mathcal{Q}_{F_\psi}$ is as follows: Given an instance of CNF formula represented as $\mathbf{A} = (A, C, P, N)$, with universe A, construct the structure $\mathbf{B} = (V \cup A, V, C, E, P, N)$. It can be checked easily that

$$f_{\#\mathrm{CNF}}(\mathbf{A}) = f_{F_\psi}(\mathbf{B}).$$

Hence $\mathcal{Q}_{F_\psi}$ is approximable by an polynomial time (ε, δ) randomized approximation algorithm for some ε, δ, if and only if ψ is true on all finite structures.

(b) The proof is similar to that of (a). ∎

5. FEASIBILITY BEYOND $\#\Sigma_1$

We would like to see whether the classes $\#\Pi_1$ and $\#\Sigma_2$ also capture some aspect of the computational complexity of the counting problems in them; i.e., are functions in $\#\Pi_1$ (respectively, $\#\Sigma_2$) easier under some reasonable measure of computational complexity than those which are not in the class? We do not know of a direct answer to this question. One intuitive reason for the difficulty in answering this question is that $\#\Pi_1$ (and hence $\#\Sigma_2$) has counting functions which are complete for the class #P under parsimonious reductions, e.g., #3CNF. Therefore any reasonably defined complexity class which contains a complete problem, like #3CNF, contains all of #P.

5.1. *The Class* $\#\mathrm{R}\Sigma_2$

As one approach, we turn our attention to restricting the quantifier-free part of the first-order formulae to study the classes we may obtain and their computational properties. We isolate a syntactic subclass of $\#\Sigma_2$ that has complete problems and every problem in this class has a FPTRAS.

DEFINITION 5.1. Let σ be a vocabulary and let $\mathcal{Q}$ be a counting problem, with finite ordered strcutres $\mathbf{A}$ over σ as instances. We say that $\mathcal{Q}$ is in the class $\#\mathrm{R}\Sigma_2$ if there is a quantifier-free first-order formula $\psi_{\mathcal{Q}}(\mathbf{z}, \mathbf{T})$ over $\sigma \cup \mathbf{T}, \mathbf{z}$ being the sequence of free first-order variables in the formula, such that

$$f_{\mathcal{Q}}(\mathbf{A}) = |\{\langle \mathbf{T}, \mathbf{z}\rangle : \mathbf{A} \models \exists \mathbf{x}\, \forall \mathbf{y} \psi_{\mathcal{Q}}(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{T})\}|,$$

where $\psi_{\mathcal{Q}}(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{T})$ is a quantifier-free formula, and when $\psi_{\mathcal{Q}}$ is expressed by an equivalent formula in conjunctive normal form, each conjunct has at most one occurrence of a predicate symbol from $\mathbf{T}$.

LEMMA 4. #DNF *is complete for* $\#\mathrm{R}\Sigma_2$ *under product reductions.*

Proof. To see that #DNF is in $\#\mathrm{R}\Sigma_2$, it is easy to observe that the Σ_2 formula which defines the #DNF problem has the required restrictions (see Section 2). To show that #DNF is hard for the class $\#\mathrm{R}\Sigma_2$, consider a counting function $\mathcal{Q}$ in $\#\mathrm{R}\Sigma_2$ which is expressible as $f_{\mathcal{Q}}(\mathbf{A}) = |\{\langle \mathbf{T}, \mathbf{z}\rangle : \mathbf{A} \models (\exists \mathbf{x})(\forall \mathbf{y})\, \psi_{\mathcal{Q}}(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{T})\}|$, where $\psi_{\mathcal{Q}}(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{T})$ is a quantifier-free formula is conjunctive normal form, $\mathbf{T}$ is a sequence $(T_1, T_2, \ldots, T_r)$ of second-order predicate variables, and $\mathbf{z}$ is an m-tuple $(z_1, \ldots, z_m)$. Let the arity of $\mathbf{x}$ be p_1 and the arity of $\mathbf{y}$ be q_1. Let $|\mathbf{A}|^{p_1} = p$, $|\mathbf{A}|^{q_1} = q$, $A^{p_1} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_p\}$, and let $A^{q_1} = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_q\}$.

It can be seen that $\mathbf{A} \models (\exists \mathbf{x})(\forall \mathbf{y})\, \psi_{\mathcal{Q}}(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{T})$ if and only if $\mathbf{A} \models \bigvee_{i=1}^{p} \bigwedge_{j=1}^{q} \psi_{\mathcal{Q},i,j}(\mathbf{T}, \mathbf{z})$, where $\psi_{\mathcal{Q},i,j}(\mathbf{T}, \mathbf{z})$ is obtained from $\psi_{\mathcal{Q}}(\mathbf{x}_i, \mathbf{y}_j, \mathbf{z}, \mathbf{T})$ by replacing every subformula of $\psi_{\mathcal{Q}}(\mathbf{x}_i, \mathbf{y}_j, \mathbf{z}, \mathbf{T})$ that is true in $\mathbf{A}$ by the logical value TRUE and every subformula of $\psi_{\mathcal{Q}}(\mathbf{x}_i, \mathbf{y}_j, \mathbf{z}, \mathbf{T})$ that is false in $\mathbf{A}$ by the logical value FALSE. In this way, we obtain a DNF formula, say $\theta_{\mathbf{A}}$, with propositional variables of the form $T_i(\mathbf{w}_i)$, where $\mathbf{w}_i \in A^{a_i}$. This uses the fact that the formula $\psi_{\mathcal{Q}}(\mathbf{x}_i, \mathbf{y}_j, \mathbf{z}, \mathbf{T})$ has only one occurrence of any predicate from the sequence of predicates $\mathbf{T}$ per conjunct.

Let $c_{\mathbf{A}}$ be the number of propositional variables amongst those above, which do not occur in $\theta_{\mathbf{A}}$. It can be easily verified that the value of $f_{\mathcal{Q}}(\mathbf{A})$ is exactly $2^{c_{\mathbf{A}}} \times$ the number of satisfying assignments of $\theta_{\mathbf{A}}$. To conclude, note that the above reduction from $\mathbf{A}$ to $\theta_{\mathbf{A}}$ is computable in polynomial time. ∎

THEOREM 6. (a) *Every counting problem in* $\#\mathrm{R}\Sigma_2$ *has an* FPTRAS.

(b) *The decision version of every counting problem in* $\#\mathrm{R}\Sigma_2$ *is in* P.

Proof. (a) Using Proposition 2 and Lemma 4, the theorem follows from the result of Karp *et al.* [KLM89] which says that #DNF has a EPTRAS.

(b) The proof of this part follows from Proposition 1 and Lemma 4. ∎

These results are similar in flavor to that obtained by Kolaitis and Thakur [KT91] in the context of optimization problems. They isolated a class MIN $\mathrm{F}^{+}\Pi_2(1)$ of NP optimization problems by restricting the quantifier-free part

of Π_2 formulae which had MIN SET COVER as a complete problem, via a approximation-preserving reduction.

Examples of other problems in $\#R\Sigma_2$ are #NON-HITTING-SETS, #NON-DOMINATING-SETS, and #NON-EDGE-DOMINATING-SETS. By Theorem 6 these problems have a FPTRAS.

5.2. *Closure Classes*

Another approach to obtain classes of counting problems with feasible computational properties is to consider the closure of the classes $\#\Sigma_0$ and $\#\Sigma_1$ under product reductions. Then, the functions in #P form a hierarchy with three levels.

DEFINITION 5.2. Given a class χ of counting problems, the closure class $P_{pr}(\chi)$ is the class of counting problems which are product reducible to some problem in χ. In particular, we are interested in the closure classes $P_{pr}(\#\Sigma_0)$, $P_{pr}(\#\Sigma_1)$, and $P_{pr}(\#\Pi_1)$.

The following theorem states the computational properties of the closure classes of interest.

THEOREM 7. (a) *Every counting problem in the class $P_{pr}(\#\Sigma_0)$ is computable in polynomial time. In fact, $P_{pr}(\#\Sigma_0)$ is exactly the class of polynomially computable counting functions.*

(b) *Every problem in $P_{pr}(\#\Sigma_1)$ has a* FPTRAS.

(c) *The decision version of every problem in $P_{pr}(\#\Sigma_1)$ is in* P.

Proof. (a) The forward direction follows from Theorem 3 and Proposition 1. The converse follows trivially.

(b) The proof follows from Theorem 4 and Proposition 2.

(c) The proof follows from the definition of product reduction. ∎

We now study the relationship between the closure classes $P_{pr}(\#\Sigma_0)$, $P_{pr}(\#\Sigma_1)$, and $P_{pr}(\#\Pi_1)$. The following theorem shows that although these classes form a hierarchy with three levels, it is very hard to prove that the classes are distinct. The distinctness of these classes has complexity theoretic consequences, as shown below.

THEOREM 8. (a) *The classes $P_{pr}(\#\Sigma_0)$, $P_{pr}(\#\Sigma_1)$, and $P_{pr}(\#\Pi_1)$ form a hierarchy with three levels,*

$$P_{pr}(\#\Sigma_0) \subseteq P_{pr}(\#\Sigma_1) \subseteq P_{pr}(\#\Pi_1) = \#P,$$

and their distinctness has consequences as below.

(b) $P_{pr}(\#\Sigma_0) = P_{pr}(\#\Sigma_1)$ *if and only if* $P = P^{\#P}$.

(c) *If* $P_{pr}(\#\Sigma_1) = P_{pr}(\#\Pi_1)$ *then* NP = RP.

Proof. (a) The containments follow from Theoerem 2. The last equality follows from the fact that #3CNF, a problem complete for #P under parsimonious reductions, is in $\#\Pi_1$.

(b) To show the forward direction, it suffices to note that $\#\Sigma_1$ has #P-complete problems, for example, #3DNF. The other direction follows trivially.

(c) The proof of this part follows from Proposition 1 and Theorem 4 and the fact that if #CNF has a polynomial time randomized approximation algorithm, then NP = RP. ∎

It should be pointed out that a similar situation holds with problems in the class NP. For example, 3COLORABILITY is expressible using a *strict* Σ_1^1 formula, i.e., an existential second-order formula whose first-order part is universal. This is a probably strict subclass of Σ_1^1 while the closure of strict Σ_1^1 under polynomial reducibilities is the entire class NP.

6. CONCLUDING REMARKS

This work has initiated logic-based investigations of the function class #P and has obtained some positive results. We discuss below some unresolved issues which either arise from this work, or are more naturally expressible in this setting.

As noted already, counting problems like #3CNF and #CNF, which are parsimoniously complete for #P, are not likely to be in the class $\#\Sigma_1$ or its closure $P_{pr}(\#\Sigma_1)$, unless there are unlikely complexity theoretic collapses such as NP = RP (see Remark 6). However, there are several other #P-complete problems which are not members of $\#\Sigma_1$, but may still be in $P_{pr}(\#\Sigma_1)$ without seemingly having any drastic complexity theory consequences. For example, is #2CNF (the restriction of #CNF with at most two literals per clause) product reducible to #3DNF or #DNF? More generally, are counting problems, whose decision versions known to be in P, product reducible to #DNF? Such reductions, if they exist, would give polynomial time randomized approximation algorithms for problems outside $\#\Sigma_1$ or $\#R\Sigma_2$. Even though such reductions do not seem to have any unlikely complexity theoretic consequence, we believe they are unlikely. For example, we conjecture that there is no product reduction from #2CNF to #DNF. We also believe that the answer to this conjecture will come from finer logic-based classification of the functions in #P.

As mentioned in Section 5, we would like to know if the class $\#\Pi_1$ (and $\#\Sigma_2$) also captures counting problems which are feasible under some other notion of feasibility. Any such feasibility notion which separates $\#\Pi_1$ from the rest of #P would essentially separate the difficulty of #3CNF from #CNF and such an answer would be interesting.

There are some counting problems in #P for which there are efficient superpolynomial time approximation algorithms or for which there are polynomial time randomized approximation algorithms for special cases of inputs (see, for example, [DLMV88]). We have not dealt with such cases in this paper. It will be interesting to see if in these cases too, logical definability of the problems has a role to play. The techniques used in this paper may also be applicable to other counting classes, like Span-P, rank-P, etc. [KST89].

Finally, we would like a syntactic characterization of problems like #EXT which are not in $\#\Sigma_1$ but which have polynomial time randomized approximation algorithms [DFK91].

ACKNOWLEDGMENTS

We thank J. Radhakrishnan for assisting us with part 4 of the proof of Theorem 2 and C. Papadimitriou for suggesting that we study the closure classes. We also thank Phokion Kolaitis and Phil Long for reading the paper thoroughly and making suggestions that vastly improved the readability of the paper. The last author would also like to thank Phokion Kolaitis for his encouragement and support.

REFERENCES

[BIS90] D. M. Barrington, N. Immerman, and H. Straubing, On uniformity within NC, *J. Comput. System Sci.* **41**, No. 3 (1990), 274–306.

[CH89] J. Y. Cai and L. Hemachandra, Enumerative counting is hard, *Inform. and Control* **82**, No. 1 (1989), 34–44.

[DFK91] M. Dyer, A. Frieze, and R. Kannan, A random polynomial time algorithm for approximating the volume of a convex body, *J. Assoc. Comput. Mach.* **38**, No. 1 (1991), 1–17.

[DLMV88] P. Dagum, M. Luby, M. Mihail, and U. Vazirani, Polytopes, permanents and graphs with large facts, *in* "Proceedings 29th IEEE Symp. on Foundations of Computer Science, 1988," pp. 412–421.

[Fag74] R. Fagin, Generalized first-order spectra and polynomial time recognizable sets, *in* "Complexity of Computations, SIAM-AMS Proc. 7, 1974" (R. Karp, Ed.), pp. 43–73.

[Gur88] Y. Gurevich, Logic and the challenge of computer science, *in* "Trends in Theoretical Computer Science" (E. Börger, Ed.), pp. 1–57, Comput. Sci. Press, Rockville, MD, 1988.

[HR90] T. Hagerup and C. Rüb, A guided tour of chernoff bounds, *Inform. and Control* **33**, No. 6 (1990), 305–308.

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

[Imm87a] N. Immerman, Expressibility as a complexity measure: Results and directions, *in* "Proceedings, Structure in Complexity Theory, Second Annual Conference, 1987," pp. 194–202.

[Imm87b] N. Immerman, Languages that capture complexity classes, *SIAM J. Comput.* **16** (1987), 760–778.

[Imm89] N. Immerman, Descriptive and computational complexity, *in* "Computational Complexity Theory" (J. Hartmanis, Ed.), Proceedings, AMS Symposia in Applied Mathematics, Vol. 38, Amer. Math. Soc., Providence, RI, 1989.

[KLM89] R. M. Karp, M. Luby and N. Madras, Monte carlo algorithms for enumeration and reliability problems, *J. Algorithms* **10**, No. 3 (1989), 429–448; *in* "Proceedings, 24th IEEE Symp. on Foundations of Computer Science, 1983."

[KST89] J. Köbler, U. Schöning, and J. Toran, On counting an approximation, *Acta Inform.* **26**, No. 4 (1989), 363–379.

[KT91] Ph. G. Kolaitis and M. N. Thakur, Approximaztion properties of NP minimization classes, *in* "Proceedings, Structure in Complexity Theory, Sixth Annual Conference," pp. 353–366, 1991; *J. Comput. System Sciences*, to appear.

[KT93a] Ph. G. Kolaitis and M. N. Thakur, "Logical definability of NP optimization problems," Technical Report UCSC-CRI-93-10, Computer and Information Sciences, University of California, Santa Cruz, 1993; *Inform. and Comput.*, to appear.

[KT93b] Ph. G. Kolaitis and M. N. Thakur, Polynomial time optimization, parallel approximation and fixpoint logic, *in* "Proc. Structure in Complexity Theory, Eighth Annual Conference, 1993," to appear.

[PY91] C. Papadimitriou and M. Yannakakis, Optimization, approximation and complexity classes, *J. Comput. System Sci.* **43**, No. 3 (1991), 425–440; *in* "Proceedings, 20th Annual ACM Symp. on Theory of Computing, 1988."

[R70] A. Rényi, "Probability Theory," North-Holland, Amsterdam, 1970.

[Sto76] L. J. Stockmeyer, The polynomial time hierarchy, *Theoret. Comput. Sci.* **3** (1976), 1–22.

[Sto85] L. Stockmeyer, On approximation algorithms for #P, *SIAM J. Comput.* **14** (1985), 849–861; *in* "Proceedings, 15th ACM Symp. on Theory of Computing, 1983."

[Tod89] S. Toda, On the computational power of PP and $\oplus$P, *in* "Proceedings, 30th IEEE Symp. on Foundations of Computer Science, 1989," pp. 514–519.

[Val79] L. G. Valiant, The complexity of computing the permanent, *Theoret. Comput. Sci.* **8** (1979), 189–201.

[Var82] M. Y. Vardi, The complexity of relational query languages, *in* "Proceedings, 14th ACM Symp. on Theory of Computing, 1982," pp. 137–146.

[VV86] L. G. Valiant and V. V. Vazirani, NP is as easy as detecting unique solutions, *Theoret. Comput. Sci.* **47** (1986), 85–93.