Counting of Teams in First-Order Team Logics

Anselm Haak
Theoretische Informatik, Leibniz Universität Hannover, Appelstraße, D-30167, Germany
haak@thi.uni-hannover.de

Juha Kontinen
Department of Mathematics and Statistics, University of Helsinki,
Pietari Kalmin katu 5, 00014, Finland
juha.kontinen@helsinki.fi

Fabian Müller
Theoretische Informatik, Leibniz Universität Hannover, Appelstraße, D-30167, Germany
fabian.mueller@thi.uni-hannover.de

Heribert Vollmer
Theoretische Informatik, Leibniz Universität Hannover, Appelstraße, D-30167, Germany
vollmer@thi.uni-hannover.de

Fan Yang
Department of Mathematics and Statistics, University of Helsinki,
Pietari Kalmin katu 5, 00014, Finland
fan.yang@helsinki.fi

Abstract

We study descriptive complexity of counting complexity classes in the range from #P to $\# \cdot \mathrm{NP}$. A corollary of Fagin's characterization of NP by existential second-order logic is that #P can be logically described as the class of functions counting satisfying assignments to free relation variables in first-order formulae. In this paper we extend this study to classes beyond #P and extensions of first-order logic with team semantics. These team-based logics are closely related to existential second-order logic and its fragments, hence our results also shed light on the complexity of counting for extensions of first-order logic in Tarski's semantics. Our results show that the class $\# \cdot \mathrm{NP}$ can be logically characterized by independence logic and existential second-order logic, whereas dependence logic and inclusion logic give rise to subclasses of $\# \cdot \mathrm{NP}$ and #P, respectively. We also study the function class generated by inclusion logic and relate it to the complexity class $\mathrm{TotP} \subsetneq \#\mathrm{P}$. Our main technical result shows that the problem of counting satisfying assignments for monotone Boolean Σ_1-formulae is $\# \cdot \mathrm{NP}$-complete with respect to Turing reductions as well as complete for the function class generated by dependence logic with respect to first-order reductions.

2012 ACM Subject Classification Theory of computation → Models of computation; Theory of computation → Complexity classes; Theory of computation → Complexity theory and logic

Keywords and phrases team-based logics, counting classes, finite model theory, descriptive complexity

Digital Object Identifier 10.4230/LIPIcs.MFCS.2019.19

Related Version A full version of the paper is available at `https://arxiv.org/abs/1902.00246`.

Funding *Anselm Haak, Fabian Müller, Heribert Vollmer*: Supported by DFG VO 630/8-1 and DAAD grant 57348395.
Juha Kontinen, Fan Yang: Supported by grants 308712 and 308099 of the Academy of Finland.

Acknowledgements We thank the anonymous referees for helpful comments.

1 Introduction

The question of the power of counting arises in propositional and predicate logic in a number of contexts. Counting the number of satisfying assignments for a given propositional formula, #SAT, is complete for Valiant's class #P of functions counting accepting paths of nondeterministic polynomial-time Turing machines [32]. Valiant also proved that #SAT even remains complete when restricted to monotone 2CNF-formulae.

The class #P can be seen as the counting analogue of NP, which was shown by Fagin [11] to correspond to existential second order logic, where the quantified relation encodes accepting computation paths of NP-machines. Hence, if we define $\#\mathrm{FO}^{\mathrm{rel}}$ to count satisfying assignments to free relational variables in first-order (FO-) formulae, we obtain $\#\mathrm{FO}^{\mathrm{rel}} = \#\mathrm{P}$. This result has been refined to prefix classes of FO showing, e.g., that $\#\Pi_2^{\mathrm{rel}} = \#\mathrm{P}$ [30].

If we define $\#\mathrm{FO}^{\mathrm{func}}$ in the same fashion as $\#\mathrm{FO}^{\mathrm{rel}}$ except that we count assignments to function variables instead of relation variables, then obviously $\#\mathrm{FO}^{\mathrm{func}} = \#\mathrm{P}$. The situation changes for the prefix classes, though. In particular, unlike for $\#\Pi_1^{\mathrm{rel}}$, it holds that $\#\Pi_1^{\mathrm{func}} = \#\mathrm{P}$, and, remarkably, also arithmetic circuit classes like $\#\mathrm{AC}^0$ can be characterized in this context [7].

In this paper we consider a different model-theoretic approach to the study of counting processes using so-called team-based logics. In these logics, formulae with free variables are evaluated not for single assignments to these variables but for *sets* of such assignments (called *teams*). Logics with team semantics have been developed for the study of various dependence and independence concepts important in many areas such as (probabilistic) databases and Bayesian networks (see, e.g. [16, 5, 15]) for which model counting is an important inference task (see, e.g., [3, 27]). In addition, team-based logics have interesting connections to a wide range of areas such as formal semantics of natural language [4], social choice theory [28], and quantum information theory [18].

In team semantics, a first-order formula is satisfied by a team if and only if all its members satisfy the formula individually. Interest in teams stems from the introduction of different logical atoms describing properties of teams, called team atoms, such as the value of a variable being functionally dependent on other variables (characterized by the *dependence* atom $=(\bar{x}, y)$), a variable being independent from other variables (characterized by the *independence* atom $\bar{y} \perp_{\bar{x}} \bar{z}$), and the values of a variable occurring as values of some other variable (characterized by the *inclusion* atom $\bar{x} \subseteq \bar{y}$), etc. ([31, 14, 12]).

We initiate in this paper the study of counting for team-based logics. In our proofs we utilize the known correspondences between team-based logics and existential second-order logic (Σ_1^1) and its fragments (see Theorem 2). We want to stress that our results are also novel for existential second-order logic and its fragments, and that there is no natural way to carry out the study of the function class generated by inclusion logic, that is, FO extended by the inclusion atom, in Tarskian semantics.

We define $\#\mathrm{FO}^{\mathrm{team}}$ to be the class of functions counting teams that satisfy a given FO-formula, and similarly for extensions of FO by team atoms. Making use of different team atoms, we give a characterization of $\# \cdot \mathrm{NP}$. While it is relatively easy to see that with every finite set A of NP-definable team atoms, the class $\#\mathrm{FO}(A)^{\mathrm{team}}$ stays a subclass of $\# \cdot \mathrm{NP}$ (Toda's generalization of #P, see [17] for a survey of counting classes like these), we show that FO extended with the independence atom is actually sufficient to characterize the full class $\# \cdot \mathrm{NP}$:

$$\#\mathrm{FO}(\perp)^{\mathrm{team}} = \#\Sigma_1^1 = \# \cdot \mathrm{NP}.$$

The situation with inclusion logic and dependence logic is more complex due to their strong closure properties: satisfaction of formulae is closed under union for inclusion logic and is closed downwards for dependence logic. We show that $\#\mathrm{FO}(\subseteq)^{\mathrm{team}}$ is a subclass of TotP, which is a class of counting problems with easy decision versions. Note that TotP is a strict subclass of #P unless P = NP and consequently the same holds for $\#\mathrm{FO}(\subseteq)^{\mathrm{team}}$. Furthermore, $\#\mathrm{FO}(=(\dots))^{\mathrm{team}}$ is a subclass of $\# \cdot \mathrm{NP}$, which we believe to be strict as well. Interestingly, both classes contain complete problems from their respective superclasses. In establishing this result for dependence logic, we introduce an interesting class of monotone quantified Boolean formulae and show that the corresponding counting problem where the all-0-assignment is not counted, $\#\Sigma_1\mathrm{CNF}^{-}_{*}$, is $\# \cdot \mathrm{NP}$-complete. In order to prove $\# \cdot \mathrm{NP}$-completeness we also show that the more natural problem of counting all satisfying assignments of the same class of formulae is $\# \cdot \mathrm{NP}$-complete by introducing a new technique of simultaneous reductions between pairs of counting problems, which we hope will also be useful in other contexts.

For inclusion logic we show that the well-known #P-complete problem $\#2\mathrm{CNF}^{+}$ is in $\#\mathrm{FO}(\subseteq)^{\mathrm{team}}$ and that the problem of counting assignments for existentially quantified dual-Horn formulae is hard for $\#\mathrm{FO}(\subseteq)^{\mathrm{team}}$.

In related previous work, so-called weighted logics have been used to logically characterize counting complexity classes [1], and the decision-problem analogue PP of #P and the counting hierarchy have been logically characterized in [21, 23, 6]. Counting classes from circuit complexity beyond $\#\mathrm{AC}^0$ have been logically characterized in [8].

Due to space restrictions, we only give proof sketches for some theorems, and detailed proofs for all results throughout the paper are deferred to the full version of this paper.

2 Definitions and Preliminaries

First-order Logic and Team Semantics

Let us start by recalling the syntax of first-order logic (FO). In this work, we only consider relational vocabularies (i.e., vocabularies with no function or constant symbols), and thus the only first-order terms are variables. Formulae of first-order logic are defined by the following grammar:

$$\varphi ::= \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \exists x \varphi \mid \forall x \varphi \mid R(\overline{x}) \mid \neg R(\overline{x}) \mid x = y \mid x \neq y \qquad (\star)$$

where x, y are variables, R is a relation symbol, and $\overline{x}$ is a tuple of the appropriate number of variables.

The set of *free variables* of a formula φ is defined as usual, and we sometimes write $\varphi(x_1, \dots, x_k)$ to emphasize that the free variables of φ are among $x_1, \dots, x_k$. A formula with no free variable is called a *sentence*. For any k, the fragment Σ_k of FO consists of all formulae of the form $\exists\overline{x_1}\forall\overline{x_2}\dots Q\overline{x_k}\varphi$, where φ is quantifier-free and Q is either $\exists$ or $\forall$ depending on whether k is odd or even; similarly, the fragment Π_k is defined as the class of all formulae $\forall\overline{x_1}\exists\overline{x_2}\dots Q\overline{x_k}\varphi$ in prenex normal form with a quantifier prefix with k alternations starting with universal quantifiers.

We only consider finite structures with a finite relational vocabulary σ. Denote the class of all such structures by $\mathrm{STRUC}[\sigma]$, and let $\mathrm{dom}(\mathcal{A})$ denote the universe of a σ-structure $\mathcal{A}$. We will always use structures with universe $\{0, 1, \dots, n-1\}$ for some $n \in \mathbb{N} \setminus \{0\}$. We assume that our structures contain a *built-in binary relation* $\leq$ and *ternary relations* $+, \times$ with the usual interpretation, i.e., $\leq$ is interpreted in a model of any size as the "less than or equal to" relation on $\mathbb{N}$, $+$ is interpreted as addition and $\times$ as multiplication. We write $\mathrm{enc}_\sigma(\mathcal{A})$ for the standard binary encoding of a σ-structure $\mathcal{A}$ (see e.g., [20]): Relations are encoded row by row by listing their truth values as 0's and 1's. The whole structure is encoded by the concatenation of the encodings of its relations.

We assume that the reader is familiar with the usual Tarskian semantics for first-order formulae, in which formulae are evaluated with respect to single assignments of a structure. In this paper, we also consider so-called *team semantics* for first-order formulae, in which formulae are evaluated with respect to teams. A *team* is a *set* of assignments of a structure, that is, a set of functions $s\colon \{x_1, \dots, x_k\} \to \mathrm{dom}(\mathcal{A})$, where we call $\{x_1, \dots, x_k\}$ the domain of the team. Note that the empty set $\emptyset$ is a team, called *empty team*, and the singleton $\{\emptyset\}$ containing only the empty assignment is also a team. We denote by $\mathrm{team}(\mathcal{A}, (x_1, \dots, x_k))$ the set of all teams over a structure $\mathcal{A}$ with the domain $\{x_1, \dots, x_k\}$. Due to certain connections between team logics and second-order logic it is often helpful to view teams as relations. Let $\mathcal{A}$ be a structure and X a team of $\mathcal{A}$ with domain $\{x_1, \dots, x_k\}$. $\mathcal{A}$ and X induce the k-ary relation $\mathrm{rel}(X)$ on $\mathrm{dom}(\mathcal{A})$ defined as

$$\mathrm{rel}(X) := \{(s(x_1), \dots, s(x_n)) \mid s \in X\}.$$

Furthermore, for any subset $V \subseteq \{x_1, \dots, x_k\}$ of variables we define

$$X\big|_V := \{s\big|_V \mid s \in X\},$$

the restriction of team X to domain V.

We define inductively the notion of a team X with domain $\{x_1, \dots, x_k\}$ of a structure $\mathcal{A}$ with $A := \mathrm{dom}(\mathcal{A})$ satisfying an FO-formula $\varphi(x_1, \dots, x_k)$, denoted by $\mathcal{A} \models_X \varphi$, as follows:

- $\mathcal{A} \models_X \alpha$ for α an atomic formula if and only if for all $s \in X$, $\mathcal{A} \models_s \alpha$ in the usual Tarskian semantics sense.
- $\mathcal{A} \models_X \varphi \vee \psi$ if and only if there are teams $Y, Z \subseteq X$ such that $Y \cup Z = X$, $\mathcal{A} \models_Y \varphi$ and $\mathcal{A} \models_Z \psi$.
- $\mathcal{A} \models_X \varphi \wedge \psi$ if and only if $\mathcal{A} \models_X \varphi$ and $\mathcal{A} \models_X \psi$.
- $\mathcal{A} \models_X \exists x\varphi$ if and only if there exists a function $F\colon X \to \mathcal{P}(A) \setminus \{\emptyset\}$, called *supplementing function*, such that $\mathcal{A} \models_{X[F/x]} \varphi$, where

$$X[F/x] = \{s[a/x] \mid s \in X \text{ and } a \in F(s)\} \qquad \text{and} \quad s[a/x](y) = \begin{cases} a, & \text{if } x = y, \\ s(y), & \text{else.} \end{cases}$$

- $\mathcal{A} \models_X \forall x\varphi$ if and only if $\mathcal{A} \models_{X[A/x]} \varphi$, where $X[A/x] = \{s[a/x] \mid s \in X \text{ and } a \in A\}$.

A sentence φ is said to be *true* in $\mathcal{A}$, written $\mathcal{A} \models \varphi$, if $\mathcal{A} \models_{\{\emptyset\}} \varphi$.

FO-formulae φ are flat over team semantics, i.e., $\mathcal{A} \models_X \varphi$, if and only if $\mathcal{A} \models_s \varphi$ for all $s \in X$. In this sense, team semantics is conservative over FO-formulae. We now extend first-order logic by sets of atomic formulae which are not flat. For any sequence $\overline{x}$ of variables and variable y, the string $=\!(\overline{x}, y)$ is called a *dependence atom.* For any sequences $\overline{x}, \overline{y}, \overline{z}$ of variables, the string $\overline{y} \perp_{\overline{x}} \overline{z}$ is called an *independence atom.* For any two sequences $\overline{x}$ and $\overline{y}$ of variables of the same length, the string $\overline{x} \subseteq \overline{y}$ is called an *inclusion atom.* The team semantics of these atoms is defined as follows:

- $\mathcal{A} \models_X =\!(\overline{x}, y)$, if and only if for all $s, s' \in X$, if $s(\overline{x}) = s'(\overline{x})$, then $s(y) = s'(y)$.
- $\mathcal{A} \models_X \overline{y} \perp_{\overline{x}} \overline{z}$ if and only if for all $s, s' \in X$ such that $s(\overline{x}) = s'(\overline{x})$, there exists $s'' \in X$ such that $s''(\overline{x}) = s(\overline{x})$, $s''(\overline{y}) = s(\overline{y})$ and $s''(\overline{z}) = s'(\overline{z})$.
- $\mathcal{A} \models_X \overline{x} \subseteq \overline{y}$ if and only if for all $s \in X$ there is $s' \in X$ such that $s(\overline{x}) = s'(\overline{y})$.

For any subset $A \subseteq \{=(\dots), \perp, \subseteq\}$, we define $\mathrm{FO}(A)$ as first-order logic extended by the respective atoms, and refer to such a logic as *team-based logic*. More precisely we extend the grammar $(\star)$ by adding a rule for each atom in A. For example for $\mathrm{FO}(\{\subseteq\})$ we add the rule

$$\varphi ::= \overline{x} \subseteq \overline{y},$$

where $\overline{x}, \overline{y}$ are tuples of variables. For convenience, we often omit the curly brackets and write for example $\mathrm{FO}(\subseteq)$ instead of $\mathrm{FO}(\{\subseteq\})$.

The team-based logic $\mathrm{FO}(=(\dots))$ is known in the literature as *dependence logic* [31], $\mathrm{FO}(\perp)$ as *independence logic* [14] and $\mathrm{FO}(\subseteq)$ as *inclusion logic* [12]. We recall some basic properties of these logics from [31, 14, 12]: Formulae of $\mathrm{FO}(=(\dots))$ are *closed downwards*, i.e., $\mathcal{A} \models_X \varphi$ and $Y \subseteq X$ implies $\mathcal{A} \models_Y \varphi$, formulae of $\mathrm{FO}(\subseteq)$ are *closed under unions*, i.e., $\mathcal{A} \models_X \varphi$ and $\mathcal{A} \models_Y \varphi$ implies $\mathcal{A} \models_{X \cup Y} \varphi$, and formulae of any of these logics have the *empty team property*, i.e., $\mathcal{A} \models_\emptyset \varphi$ always holds.

The above atoms expressing team properties can be generalized, as we will do next. Let us first recall below the definition of generalized quantifiers, where we follow the notations from [22, 26].

▶ **Definition 1.** *Let $i_1, \dots, i_n$ $(n > 0)$ be a sequence of positive integers, and σ a vocabulary consisting of an i_j-ary relation symbol for each $1 \leq j \leq n$. A* generalized quantifier *of type $(i_1, \dots, i_n)$ is a class $\mathcal{C}$ of σ-structures $(A, B_1, \dots, B_n)$ such that the following conditions hold:*

1. *$A \neq \emptyset$ and for each $1 \leq j \leq n$, we have $B_j \subseteq A^{i_j}$.*
2. *$\mathcal{C}$ is closed under isomorphisms, that is, if $(A', B'_1, \dots, B'_n) \in \mathcal{C}$ is isomorphic to $(A, B_1, \dots, B_n)$, then $(A', B'_1, \dots, B'_n) \in \mathcal{C}$.*

Let Q be a generalized quantifier of type $(i_1, \dots, i_n)$. Let us extend the syntax of first-order logic with an expression $A_Q(\overline{x_1}, \dots, \overline{x_n})$, where each $\overline{x_j}$ is a tuple of variables of length i_j and $Vars(\overline{x_i})$ is the set of variables in x_i. We call A_Q a generalized (dependency) atom *(of type $(i_1, \dots, i_n)$), and its team semantics is defined as:*

$$\mathcal{A} \models_X A_Q(\overline{x_1}, \dots, \overline{x_n}) \textit{ if and only if} (\mathrm{rel}(X|_{Vars(\overline{x_1})}), \dots, \mathrm{rel}(X|_{Vars(\overline{x_n})})) \in Q^{\mathcal{A}},$$

where $Q^{\mathcal{A}} = \{(B_1, \dots, B_n) \mid (\mathrm{dom}(\mathcal{A}), B_1, \dots, B_n) \in Q\}$.

We say that a generalized dependency atom A_Q is NP-definable if there is an NP-algorithm that decides for a given structure $\mathcal{A}$ and a given team X whether $\mathcal{A} \models_X A_Q(\overline{x_1}, \dots, \overline{x_n})$ holds or not. A set A of generalized atoms is NP-definable if every $a \in A$ is NP-definable. For example, the set $A = \{=(\dots), \perp, \subseteq\}$ is NP-definable.

Many results in this paper are based on the expressive power of the logics defined above, that we shall now recall. We first recall some notions and notations. Existential second-order logic (Σ^1_1) consists of formulae of the form $\exists R_1 \dots \exists R_k \varphi$, where φ is an FO-formula. Let σ be a vocabulary. We write $\sigma(R)$ for the vocabulary that arises by adding a fresh relation symbol R to σ, and we sometimes write $\varphi(R)$ to emphasize that the relation symbol R occurs in the $\sigma(R)$-formula φ. If $\mathcal{A}$ is a σ-structure, we write $(\mathcal{A}, Q)$ for $\mathcal{A}$ expanded into a $\sigma(R)$-structure where the new k-relation symbol R is interpreted as $Q \subseteq \mathrm{dom}(\mathcal{A})^k$. A $\sigma(R)$-sentence $\varphi(R)$ of Σ^1_1 is said to be *downward monotone* with respect to R if $(\mathcal{A}, Q) \models \varphi(R)$ and $Q' \subseteq Q$ imply $(\mathcal{A}, Q') \models \varphi(R)$. It is known that $\varphi(R)$ is downward monotone with respect to R if and only if $\varphi(R)$ is equivalent to a sentence where R occurs only negatively (see e.g., [24]).

▶ **Theorem 2** (see [12, 24, 13]).

1. *For every σ-formula φ of* $\mathrm{FO}(\perp)$*, there is an $\sigma(R)$-sentence $\psi(R)$ of Σ_1^1 such that for all σ-structures $\mathcal{A}$ and teams X,*

$$\mathcal{A} \models_X \varphi \iff (\mathcal{A}, \mathrm{rel}(X)) \models \psi(R). \tag{1}$$

Conversely, for every $\sigma(R)$-sentence $\psi(R)$ of Σ_1^1, there is a σ-formula φ of $\mathrm{FO}(\perp)$ *such that (1) holds for all σ-structures $\mathcal{A}$ and* non-empty *teams X.*

2. *The same as the above holds for formulae of* $\mathrm{FO}(=(\dots))$ *as well, except that in both directions for* $\mathrm{FO}(=(\dots))$ *the relation symbol R is assumed to occur only negatively in the sentence $\psi(R)$.*
3. *In particular, over sentences both* $\mathrm{FO}(\perp)$ *and* $\mathrm{FO}(=(\dots))$ *are expressively equivalent to Σ_1^1, in the sense that every σ-sentence of* $\mathrm{FO}(\perp)$ *(or* $\mathrm{FO}(=(\dots))$*) is equivalent to a σ-sentence ψ of Σ_1^1, i.e., for any σ-structure $\mathcal{A}$,*

$$\mathcal{A} \models \varphi \iff \mathcal{A} \models \psi,$$

and vice versa. As a consequence of Fagin's Theorem (see [11]), over finite structures both $\mathrm{FO}(\perp)$ *and* $\mathrm{FO}(=(\dots))$ *capture* NP*.*

4. *For any σ-formula $\varphi(x_1, \dots, x_k)$ of* $\mathrm{FO}(\subseteq)$*, there exists a $\sigma(R)$-formula $\psi(R)$ of positive greatest fixed point logic (*posGFP*) such that for all σ-structures $\mathcal{A}$ and teams X,*

$$\mathcal{A} \models_X \varphi \iff \mathcal{A}, \mathrm{rel}(X) \models_s \psi(R) \textit{ for all } s \in X;$$

and vice versa. In particular, over sentences $\mathrm{FO}(\subseteq)$ *is expressively equivalent to* posGFP*. As a consequence of Immerman's Theorem (see [19]), over finite structures,* $\mathrm{FO}(\subseteq)$ *is expressively equivalent to least fixed point logic (*LFP*). Thus, by [19, 34], over ordered finite structures,* $\mathrm{FO}(\subseteq)$ *captures* P*.*

5. *Let $\varphi(R)$ be a* myopic *σ-formula, that is, $\varphi(R) = \forall \overline{x}(R(\overline{x}) \to \psi(R, \overline{x}))$, where ψ is a first order σ-formula with only positive occurrences of R. Then there exists a σ-formula $\chi \in \mathrm{FO}(\subseteq)$ such that for all σ-structures $\mathcal{A}$ and all teams X:*

$$\mathcal{A} \models_X \chi(\overline{x}) \Leftrightarrow \mathcal{A}, \mathrm{rel}(X) \models \varphi(R).$$

Propositional and Quantified Boolean formulae

In this paper, we will also consider certain classes of propositional and quantified Boolean formulae. As usual, we use CNF to denote the class of propositional formulae in conjunctive normal form and k-CNF to denote the class of propositional formulae in conjunctive normal form where each clause contains at most k literals. A formula in CNF is in the class DualHorn if each of its clauses contains at most one negative literal.

For a class $\mathcal{C}$ of Boolean formulae, we denote by $\Sigma_1\mathcal{C}$ the class of quantified Boolean formulae in prenex normal form with only existential quantifiers where the quantifier-free part is an element of $\mathcal{C}$.

For a class $\mathcal{C}$ of quantified Boolean formulae we denote with $\mathcal{C}^+$(resp. $\mathcal{C}^-$) the class of formulae in $\mathcal{C}$ whose free variables occur only positively (resp. negatively). For example, $\Sigma_1$3CNF$^-$ consists of all quantified Boolean formulae in prenex normal form with only existential quantifiers, where the quantifier-free part is in 3CNF and the free variables occur only negatively. Note that in Boolean formulae all variables are free.

Counting Problems and Counting Classes

This paper aims to identify model-theoretic characterizations of counting classes in terms of team-based logics. Let us now recall relevant previous results on the descriptive complexity of counting problems. We begin by defining the most important complexity classes for counting problems.

▶ **Definition 3.** *A function $f\colon \{0,1\}^* \to \mathbb{N}$ is in* #P *if there is a non-deterministic polynomial time Turing machine M such that for all inputs $x \in \{0,1\}^*$,*

$f(x)$ is the number of the accepting computation paths of M on input x.

This definition can be generalized as follows.

▶ **Definition 4.** *Let $\mathcal{C}$ be a complexity class. A function $f\colon \{0,1\}^* \to \mathbb{N}$ is said to be in $\# \cdot \mathcal{C}$ if there are a language $L \in \mathcal{C}$ and a polynomial p such that for all $x \in \{0,1\}^*$:*

$$f(x) = |\{y \mid |y| \leq p(|x|) \text{ and } (x,y) \in L\}|.$$

Obviously $\#\mathrm{P} = \# \cdot \mathrm{P}$, and it is well known that $\#\mathrm{P} \subseteq \# \cdot \mathrm{NP} \subseteq \# \cdot \mathrm{coNP} = \#\mathrm{P}^{\mathrm{NP}}$, where, under reasonable complexity-theoretic assumptions, all these inclusions are strict; see [17] for a survey of these issues.

▶ **Definition 5.** *A function $f\colon \{0,1\}^* \to \mathbb{N}$ is in* TotP *if there is a non-deterministic polynomial time Turing machine M such that for all inputs $x \in \{0,1\}^*$,*

$f(x)$ is the number of the computation paths of M on input x minus 1.

Subtracting 1 from the number of computation paths is neccessary since otherwise TotP-functions could never map to 0. In [29] it was shown that TotP is the closure with respect to parsimonious reductions of self-reducible counting problems from #P whose decision version is in P. It follows that $\mathrm{TotP} \subsetneq \#\mathrm{P}$ unless $\mathrm{P} = \mathrm{NP}$.

Next, we define the relevant logical counting classes.

▶ **Definition 6.** *A function $f\colon \{0,1\}^* \to \mathbb{N}$ is said to be in $\#\mathrm{FO}^{\mathrm{rel}}$ if there is a vocabulary σ with a built-in linear order $\leq$, and an* FO*-formula $\varphi(R_1,\ldots,R_k,x_1,\ldots,x_\ell)$ over σ with free relation variables $R_1,\ldots,R_k$ and free individual variables $x_1,\ldots,x_\ell$ such that for all σ-structures $\mathcal{A}$,*

$$f(\mathrm{enc}_\sigma(\mathcal{A})) = |\{(S_1,\ldots,S_k,c_1,\ldots,c_\ell) : \mathcal{A} \models \varphi(S_1,\ldots,S_k,c_1,\ldots,c_\ell\}|.$$

If the input of f is not of the appropriate form, we assume the output to be 0.

In the same fashion, subclasses of $\#\mathrm{FO}^{\mathrm{rel}}$, such as $\#\Sigma_k^{\mathrm{rel}}$ and $\#\Pi_k^{\mathrm{rel}}$ for arbitrary k, are defined by assuming that the formula φ in the above definition is in the corresponding fragments Σ_k and Π_k.

Recall the relationship between the above defined logical counting classes and #P:

▶ **Theorem 7** ([30]). $\#\Sigma_0^{\mathrm{rel}} = \#\Pi_0^{\mathrm{rel}} \subset \#\Sigma_1^{\mathrm{rel}} \subset \#\Pi_1^{\mathrm{rel}} \subset \#\Sigma_2^{\mathrm{rel}} \subset \#\Pi_2^{\mathrm{rel}} = \#\mathrm{FO}^{\mathrm{rel}} = \#\mathrm{P}$. *Furthermore, $\#\Sigma_0^{\mathrm{rel}} \subseteq \mathrm{FP}$.*

Complete problems (i.e., hardest problems) for counting classes have also been studied extensively. Let us now recall three reductions that are relevant in this study. Let f and h be counting problems. We say that f is *parsimoniously* reducible to h if there is a polynomial-time computable function g such that $f(x) = h(g(x))$ for all inputs x, f is *Turing* reducible to h if $f \in \mathrm{FP}^h$, and f is *metrically* reducible to h if there are polynomial-time computable functions g_1, g_2 such that $f(x) = g_2(h(g_1(x)), x)$ for all inputs x. Clearly, metric reductions are Turing reductions with one oracle query. Besides these three familiar reductions we now define another type of reductions, called *first-order reductions.* First recall that for any two vocabularies σ_1, σ_2, an *FO-interpretation* (or a *first-order query*) is a function $I : \mathrm{STRUC}[\sigma_1] \to \mathrm{STRUC}[\sigma_2]$, represented as a tuple $I = (\varphi_0, \varphi_{R_1}, \dots, \varphi_{R_\ell})$ of FO-formulae over σ_1 with k free variables, that maps any structure $\mathcal{A} \in \mathrm{STRUC}[\sigma_1]$ to another structure $I(\mathcal{A}) \in \mathrm{STRUC}[\sigma_2]$, whose domain is a subset of $\mathrm{dom}(\mathcal{A})^k$ (i.e., a set of k-ary tuples of elements in $\mathcal{A}$) defined by φ_0 and relations R_i are defined by φ_{R_i} (see [20] for detailed discussion). In the team semantics case, we also need to define how teams are transformed by the interpretation I. The value $I(X)$ is defined in a straightforward way: individual elements from $\mathcal{A}^{k\cdot m}$ in X are mapped to elements from $I(\mathcal{A})^m$ in $I(X)$, where k is the arity of tuples in the domain of the structure $I(\mathcal{A})$, and m is the arity of the team $I(X)$. Now, we define first-order reductions via FO-interpretations as follows:

▶ **Definition 8.** *Let f_1, f_2 be two functions. We say f_1 is first-order reducible or FO-reducible to f_2 (denoted $f_1 \leq_{fo} f_2$) if there are vocabularies σ_1, σ_2 with $\sigma_2 = (R_1^{a_1}, \dots, R_l^{a_l})$ and an FO-interpretation $I = (\varphi_0, \varphi_{R_1}, \dots, \varphi_{R_\ell})$, where $\varphi_0, \varphi_{R_1}, \dots, \varphi_{R_\ell}$ are* FO*-formulae over σ_1, such that for all $\mathcal{A}_1 \in \mathrm{STRUC}[\sigma_1]$, there are $k \in \mathbb{N}$ and $\mathcal{A}_2 \in \mathrm{STRUC}[\sigma_2]$ with*

$$dom(\mathcal{A}_2) = \{(x^1, \dots, x^k) \mid \mathcal{A}_1 \models \varphi_0(x^1, \dots, x^k)\}$$

and for all $i \leq k$

$$R_i((x_1^1, \dots, x_1^k), \dots, (x_{a_i}^1, \dots, x_{a_i}^k)) \Leftrightarrow \mathcal{A}_1 \models \varphi_{R_i}(x_1^1, \dots, x_1^k, \dots, x_{a_i}^1, \dots, x_{a_i}^k)$$

and $f_1(\mathrm{enc}_{\sigma_1}(\mathcal{A}_1)) = f_2(\mathrm{enc}_{\sigma_2}(\mathcal{A}_2))$.

It is often possible to find complete problems in counting classes by counting satisfying assignments for certain (quantified) Boolean formulae. Let $\mathcal{F}$ be a class of quantified Boolean formulae. Define the problem $\#\mathcal{F}$ as follows:

Problem:	$\#\mathcal{F}$
Input:	Formula $\varphi \in \mathcal{F}$
Output:	Number of satisfying assignments of φ

For example, #SAT, the function counting the number of satisfying assignments for propositional formulae, as well as its restriction #3CNF, are complete for #P under parsimonious reductions, while $\#2\mathrm{CNF}^+$ and $\#2\mathrm{CNF}^-$ are complete for #P under Turing reductions. Observe that for all Σ_1CNF-formulae φ it holds that the number of satisfying assignments is equal to that of the formula $\widetilde{\varphi}$ obtained by negating all literals in all clauses in φ. Thus, $\#\Sigma_1\mathrm{CNF}^+$ and $\#\Sigma_1\mathrm{CNF}^-$ are in a sense the same problem. In fact all our results for $\#\mathcal{C}^+$ (for a class of formulae $\mathcal{C}$) also hold for $\#\mathcal{C}^-$ and vice versa. This also holds for #Horn and #DualHorn. Note that Aziz et al [2] studied the problem $\#\Sigma_1$SAT under the name *projected model counting* and observed that it is contained in $\# \cdot \mathrm{NP}$.

Next we introduce the central class for this paper, a class of counting problems in the context of team-based logics. For any set A of generalized dependency atoms, we define $\#\mathrm{FO}(A)^{\mathrm{team}}$ to consist of those functions counting non-empty satisfying teams for FO(A)-formulae. Note that by the empty team property of dependence, independence, and inclusion logic formulae any function that counts all satisfying teams (including the empty team) could not attain the value 0.

▶ **Definition 9.** *For any set A of generalized atoms, $\#\mathrm{FO}(A)^{\mathrm{team}}$ is the class of all functions $f\colon \{0,1\}^* \to \mathbb{N}$ for which there is a vocabulary σ with a built-in linear order $\leq$ and an $\mathrm{FO}(A)$-formula $\varphi(\overline{x})$ over σ with a tuple $\overline{x}$ of free first-order variables such that for all σ-structures $\mathcal{A}$,*

$$f(\mathrm{enc}_\sigma(\mathcal{A})) = |\{X \in \mathit{team}(\mathcal{A},(\overline{x})) : X \neq \emptyset \text{ and } \mathcal{A} \models_X \varphi(\overline{x})\}|.$$

We denote by f_φ the function defined by φ.

▶ **Example 10.** As an example for how to work with team semantics in a counting context, we show that the #P-complete problem $\#2\mathrm{CNF}^+$ is contained in both $\#\mathrm{FO}(=\!(\dots))^{\mathrm{team}}$ and $\#\mathrm{FO}(\subseteq)^{\mathrm{team}}$. Let $\varphi(x_1,\dots,x_n) = \bigwedge C_i \in 2\mathrm{CNF}^+$, where each $C_i = \ell_{i,1} \vee \ell_{i,2}$ and $\ell_{i,j} \in \{x_1,\dots,x_n\}$. Consider the vocabulary $\tau_{2\mathrm{CNF}^+} = \{C^2\}$. We encode the formula $\varphi(x_1,\dots,x_n)$ by the structure $\mathcal{A} = (\{x_1,\dots,x_n\}, C^{\mathcal{A}})$, where $(x,y) \in C^{\mathcal{A}}$ if and only if the clause $x \vee y$ occurs in φ.

We show that $\#2\mathrm{CNF}^+$ can be defined by counting non-empty teams (which correspond to assignments mapping at least one variable to true) satisfying suitable formulae from $\mathrm{FO}(\subseteq)$ as well as $\mathrm{FO}(=\!(\dots))$. For this purpose, we encode Boolean assignments to the variables $x_1,\dots,x_n$ by teams over one variable t. Since the universe of our structures is exactly the set of variables of the formula $\varphi(x_1,\dots,x_k)$ in question, assignments to the variables can be encoded by inclusion of the values of the variables x_i in t in the team.

Now, the following $\mathrm{FO}(\subseteq)$-formula defines $\#2\mathrm{CNF}^+$:

$$\varphi_{\subseteq}(t) = \forall x \forall y (\neg C(x,y) \vee x \subseteq t \vee y \subseteq t).$$

Intuitively, this formula states that if a pair (x,y) of variables in φ occur in the same clause (i.e., $C(x,y)$ holds), then the value of one of the two variables x,y is contained in the team, or it is set to true.

To define the same problem in $\mathrm{FO}(=\!(\dots))$ where inclusion atoms are not any more available in the language, we need to encode assignments differently. We now encode variables x_i being set to 1 by not including them in the team over the variable t. The $\mathrm{FO}(=\!(\dots))$-formula that defines $\#2\mathrm{CNF}^+$ is the following:

$$\begin{aligned}\varphi_{=(\dots)}(t) = {} & \exists \min \forall z\ \min \leq z \wedge \\ & \quad \forall x \forall y \exists x' \exists y' \Big(=\!(x,x') \wedge =\!(y,y') \wedge (\neg x = y \vee x' = y') \wedge \\ & \qquad (x \neq t \vee x' = \min) \wedge (\neg C(x,y) \vee x' \neq \min \vee y' \neq \min) \Big).\end{aligned}$$

Intuitively, in the above formula we use existential quantifications together with dependence atoms to express that x' is a function of x, and this function f is guaranteed in the formula to be consistent with the assignment encoded by the team. We shall interpret a function mapping to the minimal element of the universe (encoded by the variable min in the formula) as an assignment to 0, and a function mapping to any other element as an assignment to 1. Now, the second last conjunct in our formula states that all variables that occur as values of t in the team (which correspond to those x_i set to 0) are mapped by our function f to the minimal element. Finally, the last conjunct in our formula checks whether the 2CNF-formula is satisfied by the assignment encoded by f. Note that in order to talk about the assignment to two variables simultaneously, in the above formula we actually use two (equivalent) functions to encode the same assignment.

3 A Characterization of the Class $\# \cdot \mathrm{NP}$

In this section, we characterize the class $\# \cdot \mathrm{NP}$ in terms of team-based logics. Our first result shows that $\# \cdot \mathrm{NP}$ is the largest class attainable by counting teams in team-based logics $\mathrm{FO}(A)$, as long as all generalized atoms in A are NP-definable.

▶ **Theorem 11.** *For any set A of* NP*-definable generalized atoms,* $\#\mathrm{FO}(A)^{\mathrm{team}} \subseteq \# \cdot \mathrm{NP}$.

Proof Sketch. Let $\varphi(\overline{x}) \in \#\mathrm{FO}(A)^{\mathrm{team}}$. To show that $f_\varphi \in \# \cdot \mathrm{NP}$ we note that f_φ can be computed by counting on input $\mathrm{enc}_\sigma(\mathcal{A})$ the number of teams X such that $\mathcal{A} \models_X \varphi(\overline{x})$. It is thus sufficient to show that the letter can be checked in NP on input $(\mathrm{enc}_\sigma(\mathcal{A}), X)$. In this proof, the only places that involve nondeterminism are disjunctions (guess the split), existential quantifiers (guess the supplementing function) and NP-definable generalized atoms (checkable in NP by definition). ◀

Next, we prove the converse inclusion of the above theorem by proving a stronger result: The whole class $\#\cdot\mathrm{NP}$ can actually be captured by a single generalized atom, the independence atom.

▶ **Theorem 12.** $\# \cdot \mathrm{NP} \subseteq \#\mathrm{FO}(\perp)^{\mathrm{team}}$

Proof Sketch. It is sufficient to show $\# \cdot \mathrm{NP} \subseteq \#\Sigma_1^1$ since by Theorem 2.1 we have $\#\Sigma_1^1 = \#\mathrm{FO}(\perp)^{\mathrm{team}}$.

Let $f \in \# \cdot \mathrm{NP}$. Then there are a polynomial p and $L \in \mathrm{NP}$ such that

$$f(x) = |\{y \mid |y| = p(|x|), (x, y) \in L\}|.$$

We encode tuples (x, y) of strings with $|y| = p(|x|)$ as structures $\mathcal{A}_{(x,y)}$ by encoding the string x as a structure $\mathcal{A}_x$ in the standard way, and y as a unique relation R_y over $\mathrm{dom}(\mathcal{A}_x)^k$ for some $k \in \mathbb{N}$. Finally, Fagin's theorem gives a Σ_1^1-sentence φ such that for all x,

$$|\{y \mid |y| = p(|x|), (x, y) \in L\}| = |\{y \mid \mathcal{A}_{(x,y)} \models \varphi\}| = |\{R \mid \mathcal{A}_x \models \varphi(R)\}|. \quad ◀$$

▶ Remark 13. The class $\#\mathrm{P}$ can also be characterized by counting teams. A variant $\mathcal{L}$ of dependence logic that defines exactly the first-order definable team properties in the sense of Theorem 2 was introduced in [25]. Since $\#\mathrm{P} = \#\mathrm{FO}$ (see [30]), this logic $\mathcal{L}$ captures $\#\mathrm{P}$. We do not present the details of $\mathcal{L}$ in this paper, but only note that $\mathcal{L}$ has weaker versions of quantifiers and disjunction instead of the standard ones as defined in Section 2.

4 Counting Teams in Dependence and Inclusion Logic

In this section, we study the smaller classes $\#\mathrm{FO}(=(\dots))^{\mathrm{team}}$ and $\#\mathrm{FO}(\subseteq)^{\mathrm{team}}$. We begin by showing that the $\# \cdot \mathrm{NP}$-complete problem $\#\Sigma_1\mathrm{CNF}^-_*$, defined below, is contained in $\#\mathrm{FO}(=(\dots))^{\mathrm{team}}$. We will show $\# \cdot \mathrm{NP}$-completeness for $\#\Sigma_1\mathrm{CNF}^-_*$ in Theorem 26.

Problem:	$\#\Sigma_1\mathrm{CNF}^-_*$
Input:	Formula $\varphi(x_1, \dots, x_k) \in \mathrm{CNF}^-$
Output:	Number of satisfying assignments of φ, disregarding the all-0-assignment

Note that the *all-0-assignment* is the assignment mapping each variable to 0.

▶ **Theorem 14.** $\#\Sigma_1\mathrm{CNF}^-_* \in \#\mathrm{FO}(=(\dots))^{\mathrm{team}}$.

We will show that the above problem is actually complete for $\#\mathrm{FO}(=(\dots))^{\mathrm{team}}$ with respect to first-order reductions. First-order reductions turn out to be particularly natural in our context, as all our classes are closed under these reductions.

▶ **Theorem 15.** $\#\mathrm{FO}(A)^{\mathrm{team}}$ *is closed under first-order reductions for* $A \subseteq \{=(\dots), \perp, \subseteq\}$.

Next we show that the problem $\#\Sigma_1\mathrm{CNF}^-_*$ is hard and thus complete for $\#\mathrm{FO}(=(\dots))^{\mathrm{team}}$ under first-order reductions. Our proof technique is similar to that of [9], where the data complexity of inclusion logic is shown to be polynomial.

▶ **Theorem 16.** $\#\Sigma_1\mathrm{CNF}^-_*$ *is complete for* $\#\mathrm{FO}(=(\dots))^{\mathrm{team}}$ *with respect to first-order reductions.*

Having proven our results for dependence logic $\mathrm{FO}(=(\dots))$, we now turn to inclusion logic $\mathrm{FO}(\subseteq)$. We first prove that $\#\mathrm{FO}(\subseteq)^{\mathrm{team}}$ is a subclass of $\#\mathrm{P}$.

▶ **Theorem 17.** $\#\mathrm{FO}(\subseteq)^{\mathrm{team}} \subseteq \#\mathrm{P}$.

The above theorem naturally gives rise to the question whether $\#\mathrm{FO}(\subseteq)^{\mathrm{team}}$ actually coincides with $\#\mathrm{P}$. However, we identify in the next lemma a particular property of $\#\mathrm{FO}(\subseteq)^{\mathrm{team}}$ functions, making this equivalence unlikely to hold.

▶ **Lemma 18.** *Let* $\varphi(\overline{x}) \in \mathrm{FO}(\subseteq)$ *be a formula over a vocabulary* σ*. Then the language* $L := \{w \mid f_\varphi(w) > 0\}$ *is in* P.

▶ **Corollary 19.** *If* $\mathrm{P} \neq \mathrm{NP}$*, then* $\#\mathrm{FO}(\subseteq)^{\mathrm{team}} \neq \#\mathrm{P}$.

Theorem 17 and Corollary 19 indicate that $\#\mathrm{FO}(\subseteq)^{\mathrm{team}}$ is most likely a strict subclass of $\#\mathrm{P}$. Nevertheless, we show in the next theorem that $\#\mathrm{FO}(\subseteq)^{\mathrm{team}}$ contains the problem #DualHorn which is complete for $\#\mathrm{P}$ with respect to Turing reductions. It is unknown whether $\#\mathrm{DualHorn} \in \#\mathrm{FO}(=(\dots))$.

▶ **Theorem 20.** $\#\mathrm{DualHorn} \in \#\mathrm{FO}(\subseteq)$.

We continue by exhibiting a hard problem for the class $\#\mathrm{FO}(\subseteq)^{\mathrm{team}}$. It is an open question whether the problem is definable by an inclusion logic formula.

▶ **Theorem 21.** $\#\Sigma_1\mathrm{DualHorn}$ *is hard for* $\#\mathrm{FO}(\subseteq)^{\mathrm{team}}$ *with respect to first-order reductions.*

It seems to us that $\#\mathrm{FO}(\subseteq)$ is a strict subclass of $\#\mathrm{P}$ and the decision versions of problems in $\#\mathrm{FO}(\subseteq)$ are in P. For this reason, we now investigate relationship between $\#\mathrm{FO}(\subseteq)$ and the subclass TotP. We show that $\#\mathrm{FO}(\subseteq)$ is a subclass of TotP and that TotP contains $\#\Sigma_1\mathrm{DualHorn}$. We conjecture that these classes do not coincide, but this question remains open.

▶ **Theorem 22.** $\#\mathrm{FO}(\subseteq) \subseteq \mathrm{TotP}$

▶ **Theorem 23.** $\#\Sigma_1\mathrm{DualHorn} \in \mathrm{TotP}$

5 Complete Problems for $\# \cdot \mathrm{NP}$

In this section we show that $\#\Sigma_1\mathrm{CNF}^-_*$ is $\# \cdot \mathrm{NP}$-complete. To this end, we first observe that $\#\Sigma_1\mathrm{CNF}$ is $\# \cdot \mathrm{NP}$-complete. Afterwards we show that the smaller class $\#\Sigma_1\mathrm{CNF}^-$ remains $\# \cdot \mathrm{NP}$-complete by adapting the proof for $\#\mathrm{P}$-completeness of $\#2\mathrm{CNF}^+$ given by Valiant [33]. We conclude this section with a reduction from $\#\Sigma_1\mathrm{CNF}^-$ to $\#\Sigma_1\mathrm{CNF}^-_*$ showing the $\# \cdot \mathrm{NP}$-completeness of the latter.

▶ **Lemma 24.** $\#\Sigma_1\mathrm{SAT}$ *and* $\#\Sigma_1\mathrm{CNF}$ *are* $\# \cdot \mathrm{NP}$*-complete under parsimonious reductions.*

▶ **Theorem 25.** $\#\Sigma_1\mathrm{CNF}^-$ *is* $\# \cdot \mathrm{NP}$*-complete under Turing reductions.*

Proof Sketch. Membership follows from 24, since $\#\Sigma_1\mathrm{CNF}^-$ is a special case of $\#\Sigma_1\mathrm{CNF}$. For the hardness proof, we show a chain of reductions adapted from the one used by Valiant [33] to show the #P-completeness of $\#2\mathrm{CNF}^+$. Recall that the main steps of Valiant's chain of reductions are:

$$\begin{aligned}\#3\mathrm{CNF} \leq \mathrm{PERMANENT} & \leq \#\mathrm{PERFECT\text{-}MATCHING}\\ \leq \#\mathrm{IMPERFECT\text{-}MATCHING} & \leq \#2\mathrm{CNF}^+.\end{aligned}$$

Our idea is to add a $\Sigma_1 3\mathrm{CNF}$-formula to the input of each problem in the above chain of reductions, and to express certain properties of the respective inputs in the added formula. We then count only the solutions to the input that also satisfy the added formula.

As part of the chain of reductions we will make use of the following problem:

Problem:	$\#(3\mathrm{CNF}, \Sigma_1 3\mathrm{CNF}^-)$
Input:	Formula $\varphi(x_1, \ldots, x_k) \in 3\mathrm{CNF}$ and formula $\psi(x_1, \ldots, x_k) \in \Sigma_1 3\mathrm{CNF}^-$
Output:	Number of satisfying assignments of $\varphi \wedge \psi$

We will reduce $\#\Sigma_1 3\mathrm{CNF}$ to $\#(3\mathrm{CNF}, \Sigma_1 3\mathrm{CNF}^-)$, and then apply the above chain of reductions (with added formulae, as described above). This results in a reduction to $\#(2\mathrm{CNF}^-, \Sigma_1 3\mathrm{CNF}^-)$, defined analogously to the above problem. Finally it is straightforward to show $\#(2\mathrm{CNF}^-, \Sigma_1 3\mathrm{CNF}^-) \leq \#\Sigma_1\mathrm{CNF}^-$ using the fact that for $\varphi \in 2\mathrm{CNF}^-$ and $\psi \in \Sigma_1 3\mathrm{CNF}^-$, the prenex normal form of $\varphi \wedge \psi$ is a $\Sigma_1 3\mathrm{CNF}^-$-formula. We conclude by sketching the first reduction.

$\underline{\#\Sigma_1 3\mathrm{CNF} \leq \#(3\mathrm{CNF}, \Sigma_1 3\mathrm{CNF}^-)}$: We construct for any $\varphi \in \Sigma_1 3\mathrm{CNF}$ two formulae $\varphi' \in 3\mathrm{CNF}$ and $\psi \in \Sigma_1 3\mathrm{CNF}^-$ such that $\#\Sigma_1 3\mathrm{CNF}(\varphi) = \#(3\mathrm{CNF}, \Sigma_1 3\mathrm{CNF}^-)(\varphi', \psi)$.

Assume $\varphi = \exists y_1 \ldots \exists y_\ell \bigwedge C_i \wedge \bigwedge D_i \wedge \bigwedge E_i$, where clauses C_i only contain free variables of φ, clauses D_i contain only bound variables of φ, and clauses E_i contain at least one free and at least one bound variable of φ. We can now simply add all clauses C_i to φ' and all clauses D_i to ψ.

To handle the remaining clauses, we add for each E_i a new free variable e_i. We then express in φ' that e_i is true if and only if clause E_i is not satisfied by the assignment to the free variables, and express in ψ that E_i has to be satisfied by the assignment to the bound variables if e_i is true. The former does not involve any bound variables and for the latter, the only needed free variable is e_i, which only occurs negatively. ◀

Because of the special role of the empty team in the team logics we consider, we will also show the completeness for another version of $\#\Sigma_1\mathrm{CNF}^-$, denoted as $\#\Sigma_1\mathrm{CNF}^-_*$, for which the all-0-assignment is not counted.

▶ **Theorem 26.** *The problem* $\#\Sigma_1\mathrm{CNF}^-_*$ *is* $\# \cdot \mathrm{NP}$*-complete under Turing reductions.*

6 Conclusion

In this paper we have studied the following hierarchy of classes defined by counting problems for team-based logics:

$$\begin{array}{ccccc}
\mathrm{TotP} & \subseteq \#\mathcal{L}^{\mathrm{team}} = \#\mathrm{P} & \subseteq & \#\mathrm{FO}(\perp)^{\mathrm{team}} = \# \cdot \mathrm{NP} \\
\cup\!\mid & & & \cup\!\mid \\
\#\mathrm{FO}(\subseteq)^{\mathrm{team}} & & & \#\mathrm{FO}(=(\dots))^{\mathrm{team}}
\end{array}$$

We also showed that our classes are closed under first-order reductions and that $\#\mathrm{FO}(\subseteq)^{\mathrm{team}}$ and $\#\mathrm{FO}(=(\dots))^{\mathrm{team}}$ contain complete problems from #P and $\# \cdot \mathrm{NP}$, respectively. The latter problem is even complete for $\#\mathrm{FO}(=(\dots))^{\mathrm{team}}$ under first-order reductions.

The connection between $\#\mathrm{FO}(=(\dots))^{\mathrm{team}}$ and the classes #P and $\# \cdot \mathrm{NP}$ is not yet clear. While we know that a complete problem from $\# \cdot \mathrm{NP}$ is contained in it, it is open whether the class coincides with $\# \cdot \mathrm{NP}$, and (if not) whether it contains the class #P. We conjecture that the answer to both questions is negative, since the defining logic has closure properties that make it unlikely to contain counting versions of non-monotone problems from #P.

Regarding $\#\mathrm{FO}(\subseteq)^{\mathrm{team}}$, the search for a complete problem could be interesting. We have only showed that the problem #DualHorn is contained in this class and the problem $\#\Sigma_1$DualHorn is hard for this class, but neither of the problems is known to be complete.

The lower end of our hierarchy deserves further study as well. The class $\#\mathrm{FO}^{\mathrm{team}}$ (i.e., the class with no dependency atoms in the formulae) can be shown to be a subclass of FTC^0, the class of functions computable by families of polynomial size constant depth majority circuits (see [35]). The circuit-based counting class $\#\mathrm{AC}^0$, counting proof trees in polynomial size constant depth unbounded fan-in circuits [35], was characterized in a model-theoretic manner by counting assignments to free function symbols in certain quantifier-restricted FO-formulae [7]. A similar quantifier restriction for $\#\mathrm{FO}(A)^{\mathrm{team}}$, where A consists of the dependency atom plus a totality atom (that we did not study in the present paper), also leads to a characterization of $\#\mathrm{AC}^0$. This suggests that low level counting classes and circuit classes in the context of counting problems for team-based logics might be worth studying. Another natural question is to search for generalized dependency atoms that lead to interesting relations to complexity classes. Besides the aforementioned totality atom, the constancy or the exclusion [12] atom are worth examining. In particular, it is an open question to find an atom A such that $\#\mathrm{FO}(A)^{\mathrm{team}} = \#\mathrm{P}$. The logic $\mathcal{L}$ of [25], though it satisfies the equality, is not of this form.

In the context of counting complexity theory, an interesting question to study is the approximability of problems in different classes. In our case, it is unlikely that any of our classes is efficiently approximable (in the sense of FPRAS): In [10] it was shown that it is unlikely that the number of satisfying assignments of CSPs can be approximated by an FPRAS, unless all relations in the constraint language are affine. Since our classes contain counting problems for classes of formulae which do not admit this property, this result applies. Consequently, it would be interesting to study restrictions of our full classes to obtain, possibly, efficiently approximable fragments.

Our proof of the completeness of $\#\Sigma_1\mathrm{CNF}^-$ for $\# \cdot \mathrm{NP}$ introduces problems that arise from "pairing decision problems" and gives simultaneous reductions between such pairs. This idea might be helpful in other contexts as well; in particular it should lead to more interesting complete problems for $\# \cdot \mathrm{NP}$ or higher levels $\# \cdot \Sigma_k$ of the counting polynomial-time hierarchy.

References

1 Marcelo Arenas, Martin Muñoz, and Cristian Riveros. Descriptive Complexity for counting complexity classes. In *LICS*, pages 1–12. IEEE Computer Society, 2017.

2 Rehan Abdul Aziz, Geoffrey Chu, Christian J. Muise, and Peter J. Stuckey. #∃SAT: Projected model counting. In *SAT*, volume 9340 of *Lecture Notes in Computer Science*, pages 121–137. Springer, 2015.

3 Fahiem Bacchus, Shannon Dalmao, and Toniann Pitassi. Solving #SAT and Bayesian Inference with Backtracking Search. *CoRR*, abs/1401.3458, 2014.

4 Ivano Ciardelli. Dependency as Question Entailment. In Samsom Abramsky, Juha Kontinen, Jouko Väänänen, and Heribert Vollmer, editors, *Dependence Logic: Theory and Application*, Progress in Computer Science and Applied Logic, pages 129–182. Birkhauser, 2016.

5 Jukka Corander, Antti Hyttinen, Juha Kontinen, Johan Pensar, and Jouko Väänänen. A Logical Approach to Context-Specific Independence. *Ann. Pure Appl. Logic*, 2019.

6 Arnaud Durand, Johannes Ebbing, Juha Kontinen, and Heribert Vollmer. Dependence Logic with a Majority Quantifier. *Journal of Logic, Language and Information*, 24(3):289–305, 2015. doi:10.1007/s10849-015-9218-3.

7 Arnaud Durand, Anselm Haak, Juha Kontinen, and Heribert Vollmer. Descriptive Complexity of $\#\mathrm{AC}^0$ Functions. In *CSL*, volume 62 of *LIPIcs*, pages 20:1–20:16. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016.

8 Arnaud Durand, Anselm Haak, and Heribert Vollmer. Model-Theoretic Characterization of Boolean and Arithmetic Circuit Classes of Small Depth. In *LICS*, pages 354–363. ACM, 2018.

9 Arnaud Durand, Juha Kontinen, Nicolas de Rugy-Altherre, and Jouko Väänänen. Tractability Frontier of Data Complexity in Team Semantics. In *GandALF*, volume 193 of *EPTCS*, pages 73–85, 2015.

10 Martin E. Dyer, Leslie Ann Goldberg, and Mark Jerrum. An approximation trichotomy for Boolean #CSP. *J. Comput. Syst. Sci.*, 76(3-4):267–277, 2010. doi:10.1016/j.jcss.2009.08.003.

11 Ronald Fagin. Generalized first-order spectra, and polynomial time recognizable sets. *SIAM-AMS Proceedings*, 7:43–73, 1974.

12 Pietro Galliani. Inclusion and exclusion dependencies in team semantics - On some logics of imperfect information. *Ann. Pure Appl. Logic*, 163(1):68–84, 2012.

13 Pietro Galliani and Lauri Hella. Inclusion Logic and Fixed Point Logic. In *CSL*, volume 23 of *LIPIcs*, pages 281–295. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2013.

14 Erich Grädel and Jouko A. Väänänen. Dependence and Independence. *Studia Logica*, 101(2):399–410, 2013.

15 Miika Hannula, Åsa Hirvonen, Juha Kontinen, Vadim Kulikov, and Jonni Virtema. Facets of Distribution Identities in Probabilistic Team Semantics. *CoRR*, abs/1812.05873, 2018.

16 Miika Hannula and Juha Kontinen. A finite axiomatization of conditional independence and inclusion dependencies. *Inf. Comput.*, 249:121–137, 2016.

17 Lane A. Hemaspaandra and Heribert Vollmer. The satanic notations: counting classes beyond #P and other definitional adventures. *SIGACT News*, 26(1):2–13, 1995.

18 Tapani Hyttinen, Gianluca Paolini, and Jouko Väänänen. Quantum Team Logic and Bell's inequalities. *Rew. Symb. Logic*, 8(4):722–742, 2015. doi:10.1017/S1755020315000192.

19 Neil Immerman. Relational Queries Computable in Polynomial Time. *Information and Control*, 68(1-3):86–104, 1986.

20 Neil Immerman. *Descriptive Complexity.* Graduate texts in computer science. Springer, 1999.

21 Juha Kontinen. A logical characterization of the counting hierarchy. *ACM Trans. Comput. Log.*, 10(1):7:1–7:21, 2009.

22 Juha Kontinen, Antti Kuusisto, and Jonni Virtema. Decidability of Predicate Logics with Team Semantics. In *MFCS*, volume 58 of *LIPIcs*, pages 60:1–60:14. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016.

23 Juha Kontinen and Hannu Niemistö. Extensions of MSO and the monadic counting hierarchy. *Inf. Comput.*, 209(1):1–19, 2011.

24 Juha Kontinen and Jouko A. Väänänen. On Definability in Dependence Logic. *Journal of Logic, Language and Information*, 18(3):317–332, 2009.

25 Juha Kontinen and Fan Yang. Logics for first-order team properties. In *WoLLIC*, Lecture Notes in Computer Science. Springer, 2019.

26 Antti Kuusisto. A Double Team Semantics for Generalized Quantifiers. *Journal of Logic, Language and Information*, 24(2):149–191, 2015.

27 Antti Kuusisto and Carsten Lutz. Weighted model counting beyond two-variable logic. In *LICS*, pages 619–628. ACM, 2018.

28 Eric Pacuit and Fan Yang. Dependence and Independence in Social Choice: Arrow's Theorem. In H. Vollmer S. Abramsky, J. Kontinen and J. Väänänen, editors, *Dependence Logic: Theory and Application*, Progress in Computer Science and Applied Logic, pages 235–260. Birkhauser, 2016.

29 Aris Pagourtzis and Stathis Zachos. The Complexity of Counting Functions with Easy Decision Version. In *MFCS*, volume 4162 of *Lecture Notes in Computer Science*, pages 741–752. Springer, 2006.

30 Sanjeev Saluja, K. V. Subrahmanyam, and Madhukar N. Thakur. Descriptive Complexity of #P Functions. *J. Comput. Syst. Sci.*, 50(3):493–505, 1995.

31 Jouko A. Väänänen. *Dependence Logic - A New Approach to Independence Friendly Logic*, volume 70 of *London Mathematical Society student texts*. Cambridge University Press, 2007.

32 Leslie G. Valiant. The Complexity of Computing the Permanent. *Theor. Comput. Sci.*, 8:189–201, 1979.

33 Leslie G. Valiant. The Complexity of Enumeration and Reliability Problems. *SIAM J. Comput.*, 8(3):410–421, 1979.

34 Moshe Y. Vardi. The Complexity of Relational Query Languages (Extended Abstract). In *STOC*, pages 137–146. ACM, 1982.

35 Heribert Vollmer. *Introduction to Circuit Complexity - A Uniform Approach*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 1999.