Complexity Thresholds in Inclusion Logic

Miika Hannula[1(✉)] and Lauri Hella[2]

[1] University of Helsinki, Helsinki, Finland
miika.hannula@helsinki.fi
[2] Tampere University, Tampere, Finland
lauri.hella@tuni.fi

Abstract. Logics with team semantics provide alternative means for logical characterization of complexity classes. Both dependence and independence logic are known to capture non-deterministic polynomial time, and the frontiers of tractability in these logics are relatively well understood. Inclusion logic is similar to these team-based logical formalisms with the exception that it corresponds to deterministic polynomial time in ordered models. In this article we examine connections between syntactical fragments of inclusion logic and different complexity classes in terms of two computational problems: maximal subteam membership and the model checking problem for a fixed inclusion logic formula. We show that very simple quantifier-free formulae with one or two inclusion atoms generate instances of these problems that are complete for (non-deterministic) logarithmic space and polynomial time. Furthermore, we present a fragment of inclusion logic that captures non-deterministic logarithmic space in ordered models.

Keywords: Team semantics · Inclusion logic · Complexity · Consistent query answering

1 Introduction

In this article we study the computational complexity of inclusion logic. Inclusion logic was introduced by Galliani [9] as a variant of dependence logic, developed by Väänänen in 2007 [26]. Dependence logic is a logical formalism that extends first-order logic with novel atomic formulae $\mathrm{dep}(x_1, \ldots, x_n)$ expressing that a variable x_n depends on variables $x_1, \ldots, x_{n-1}$. One motivation behind dependence logic is to find a unifying logical framework for analyzing dependency notions from different contexts. Since its introduction, versions of dependence logic have been formulated and investigated in a variety of logical environments, including propositional logic [16,29,31], modal logic [7,27], probabilistic logics [5], and two-variable logics [22]. Recent research has also pursued connections and applications of dependence logic to fields such as database theory [14,15],

The first author was supported by grant 308712 of the Academy of Finland.

R. Iemhoff et al. (Eds.): WoLLIC 2019, LNCS 11541, pp. 301–322, 2019.
https://doi.org/10.1007/978-3-662-59533-6_19

Bayesian networks [4], and social choice theory [24]. A common notion underlying all these endeavours is that of team semantics. Team semantics, introduced by Hodges in [17], is a semantical framework where formulae are evaluated over multitudes instead of singletons of objects as in classical logics. Depending on the application domain these multitudes may then refer to assignment sets, probability distributions, or database tables, each having their characteristic versions of team semantics [5,15,26].

After the introduction of dependence logic Grädel and Väänänen observed that team semantics can be also used to create logics for independence [11]. This was followed by [9] in which Galliani investigated logical languages built upon multiple different dependency notions. Inspired by the inclusion dependencies of database theory, one of the logics introduced was inclusion logic that extends first-order logic with inclusion atoms. Given two sequences of variables $\overline{x}$ and $\overline{y}$ having same length, an inclusion atom $\overline{x} \subseteq \overline{y}$ expresses that the set of values of $\overline{x}$ is included in the set of values of $\overline{y}$. Inclusion logic was shown to be equi-expressive to positive greatest-fixed point logic in [10]. In contrast to dependence logic which is equivalent to existential second-order logic [26], and thus to non-deterministic polynomial time (**NP**), this finding established inclusion logic as the first team-based based logic for polynomial time (**P**). Our focus in this article is to pursue this connection further by investigating the complexity of quantifier-free inclusion logic in terms of two computational problems: maximal subteam membership and model checking problems. In particular, we identify complexity thresholds for these problems in terms of first-order definability, (non-deterministic) logarithmic space, and polynomial time.

The maximal subteam membership problem $\mathsf{MSM}(\phi)$ for a formula ϕ asks whether a given assignment is in the maximal subteam of a given team that satisfies ϕ. This problem is closely related to the notion of a repair of an inconsistent database [2]. A repair of a database instance I w.r.t. some set Σ of constraints is an instance J obtained by deleting and/or adding tuples from/to I such that J satisfies Σ, and the difference between I and J is minimal according to some measure. If only deletion of tuples is allowed, J is called a subset repair. It was observed in [3] that if Σ consists of inclusion dependencies, then for every I there exists a unique subset repair J of I; this was later generalized to arbitrary LAV tgds (local-as-view tuple generating dependencies) in [25].

The research on database repair has been mainly focused on two problems: consistent query answering and repair checking. In the former, given a query Q and a database instance I the problem is to compute the set of tuples that belong to $Q(J)$ for every repair J of I. The latter is the decision problem: is J a repair of I for two given database instances I and J. The complexity of these problems for various classes of dependencies and different types of repairs has been extensively studied in the literature; see e.g. [1,3,23,25]. In this setting, the maximal subteam membership problem can be seen as a variant of the repair checking problem: regarding a team as a (unirelational) database instance I and a formula ϕ of inclusion logic as a constraint, an assignment is a positive instance of $\mathsf{MSM}(\phi)$ just in case it is in the unique subset repair of I. Note however, that

in $\mathsf{MSM}(\phi)$, the task is essentially to compute the maximal subteam from a given database instance I, instead of just checking that a given J is the unique subset repair of I. Note further, that using a single formula ϕ as a constraint is actually more general than using a (finite) set Σ of inclusion dependencies. Indeed, as ϕ we can take the conjunction of all inclusions in Σ. Furthermore, using disjunctions and quantifiers, we can form constraints not expressible in the usual formalism with a set of dependencies.

The complexity of model checking in team semantics has been studied in [6,21] for dependence and independence logics. For these logics increase in complexity arises particularly from disjunctions. For example, model checking for a disjunction of three (two, resp.) dependence atoms is complete for **NP** (**NL**, resp.), while a single dependence atom is first-order definable [21]. The results of this paper, in contrast, demonstrate that the complexity of inclusion logic formulae is particularly sensitive to conjunctions. We show that $\mathsf{MSM}(\phi)$ is complete for non-deterministic logarithmic space if ϕ is of the form $x \subseteq y$ or $x \subseteq y \wedge y \subseteq x$; for any other conjunction of (non-trivial) unary inclusion atoms $\mathsf{MSM}(\phi)$ is complete for polynomial time. This result gives a complete characterization of the maximal subteam membership problem for conjunctions of unary inclusion atoms. Based on it we also prove complexity results for model checking of quantifier-free inclusion logic formulae. For instance, for any non-trivial quantifier-free ϕ in which x, y, z do not occur, model checking of $x \subseteq y \vee \phi$ is **NL**-hard, while that of $(x \subseteq z \wedge y \subseteq z) \vee \phi$ is **P**-complete.

We conclude the paper by presenting a fragment of inclusion logic that captures **NL**. Analogous fragments have previously been established at least for dependence logic. By relating to the Horn fragment of existential second-order logic, Ebbing et al. define a fragment of dependence logic that corresponds to **P** [8]. The fragment presented in this paper is constructed by restricting occurrences of inclusion atoms and universal quantifiers, and the correspondence with **NL** is shown by using the well-known characterization of **NL** in terms of transitive closure logic [19,20].

2 Preliminaries

We generally use $x, y, z, \ldots$ for variables and $a, b, c, \ldots$ for elements of models. If $\overline{p}$ and $\overline{q}$ are two tuples, we write $\overline{pq}$ for the concatenation of $\overline{p}$ and $\overline{q}$.

Throughout the paper we assume that the reader has a basic familiarity of computational complexity. We use the notation **L**, **NL**, **P** and **NP** for the classes consisting of all problems computable in logarithmic space, non-deterministic logarithmic space, polynomial time and non-deterministic polynomial time, respectively.

2.1 Team Semantics

As is customary for logics in the team semantics setting, we assume that all formulae are in negation normal form (NNF). Thus, we give the syntax of first-

order logic (FO) as follows:

$$\phi ::= t = t' \mid \neg t = t' \mid R\bar{t} \mid \neg R\bar{t} \mid \phi \wedge \phi \mid \phi \vee \phi \mid \exists x\phi \mid \forall x\phi,$$

where t and t' are terms and R is a relation symbol of the underlying vocabulary. For a first-order formula ϕ, we denote by $\mathsf{Fr}(\phi)$ the set of free variables of ϕ, defined in the usual way. The team semantics of FO is given in terms of the notion of a *team*. Let $\mathfrak{A}$ be a model with domain A. An *assignment* s of A is a function from a finite set of variables into A. We write $s(a/x)$ for the assignment that maps all variables according to s, except that it maps x to a. For an assignment $s = \{(x_i, a_i) \mid 1 \leq i \leq n\}$, we may use a shorthand $s = (a_1, \ldots, a_n)$ if the underlying ordering $(x_1, \ldots, x_n)$ of the domain is clear from the context. A *team* X of A with domain $\mathrm{dom}(X) = \{x_1, \ldots, x_n\}$ is a set of assignments from $\mathrm{dom}(X)$ into A. For $V \subseteq \mathrm{dom}(X)$, the *restriction* $X \upharpoonright V$ of a team X is defined as $\{s \upharpoonright V \mid s \in X\}$. If X is a team, $V \subseteq \mathrm{dom}(X)$, and $F : X \to \mathcal{P}(A) \setminus \{\emptyset\}$, then $X[F/x]$ denotes the team $\{s(a/x) \mid s \in X, a \in F(s)\}$. For a set B, $X[B/x]$ is the team $\{s(b/x) \mid s \in X, b \in B\}$. Also, if s is an assignment, then by $\mathfrak{A} \models_s \phi$ we refer to Tarski semantics.

Definition 1. *For a model $\mathfrak{A}$, a team X and a formula in* FO*, the satisfaction relation $\mathfrak{A} \models_X \phi$ is defined as follows:*

- $\mathfrak{A} \models_X \alpha$ *if* $\forall s \in X : \mathfrak{A} \models_s \alpha$*, when α is a literal,*
- $\mathfrak{A} \models_X \phi \wedge \psi$ *if* $\mathfrak{A} \models_X \phi$ *and* $\mathfrak{A} \models_X \psi$*,*
- $\mathfrak{A} \models_X \phi \vee \psi$ *if* $\mathfrak{A} \models_Y \phi$ *and* $\mathfrak{A} \models_Z \psi$ *for some* $Y, Z \subseteq X$ *such that* $Y \cup Z = X$*,*
- $\mathfrak{A} \models_X \exists x\phi$ *if* $\mathfrak{A} \models_{X[F/x]} \phi$ *for some* $F : X \to \mathcal{P}(A) \setminus \{\emptyset\}$*,*
- $\mathfrak{A} \models_X \forall x\phi$ *if* $\mathfrak{A} \models_{X[A/x]} \phi$*.*

If $\mathfrak{A} \models_X \phi$, then we say that $\mathfrak{A}$ and X *satisfy* ϕ. If ϕ does not contain quantifiers or symbols from the underlying vocabulary, in which case satisfaction of a formula does not depend on the model $\mathfrak{A}$, we say that X *satisfies* ϕ, written $X \models \phi$, if $\mathfrak{A} \models_X \phi$ for all models $\mathfrak{A}$ with a suitable domain (i.e., a domain that includes all the elements appearing in X). If ϕ is a sentence, that is, a formula without any free variables, then we say that $\mathfrak{A}$ *satisfies* ϕ, and write $\mathfrak{A} \models \phi$, if $\mathfrak{A} \models_{\{\emptyset\}} \phi$, where $\{\emptyset\}$ is the team that consists of the empty assignment $\emptyset$.

We say that two sentences ϕ and ψ are equivalent, written $\phi \equiv \psi$, if $\mathfrak{A} \models \phi \iff \mathfrak{A} \models \psi$ for all models $\mathfrak{A}$. For two logics $\mathcal{L}_1$ and $\mathcal{L}_2$, we write $\mathcal{L}_1 \leq \mathcal{L}_2$ if every $\mathcal{L}_1$-sentence is equivalent to some $\mathcal{L}_2$-sentence; the relations "$\equiv$" and "$<$" for $\mathcal{L}_1$ and $\mathcal{L}_2$ are defined in terms of "$\leq$" in the standard way.

Satisfaction of a first-order formula reduces to Tarski semantics in the following way.

Proposition 2 (Flatness [26]). *For all models $\mathfrak{A}$, teams X, and formulae $\phi \in$ FO,*

$$\mathfrak{A} \models_X \phi \text{ iff } \mathfrak{A} \models_s \phi \text{ for all } s \in X.$$

A straightforward consequence is that first-order logic is downwards closed.

Corollary 3 (Downward Closure). *For all models $\mathfrak{A}$, teams X, and formulae $\phi \in$ FO,*

$$\text{If } \mathfrak{A} \models_X \phi \text{ and } Y \subseteq X, \text{ then } \mathfrak{A} \models_Y \phi.$$

2.2 Inclusion Logic

Inclusion logic (FO(⊆)) is defined as the extension of FO by inclusion atoms.

Inclusion Atom. Let $\overline{x}$ and $\overline{y}$ be two tuples of variables of the same length. Then $\overline{x} \subseteq \overline{y}$ is an *inclusion atom* with the satisfaction relation:

$$\mathfrak{A} \models_X \overline{x} \subseteq \overline{y} \text{ if } \forall s \in X \exists s' \in X : s(\overline{x}) = s'(\overline{y}).$$

Inclusion logic is *local*, meaning that satisfaction of a formula depends only on its free variables. Furthermore, the expressive power of inclusion logic is restricted by its *union closure property* which states that satisfaction of a formula is preserved under taking arbitrary unions of teams.

Proposition 4 (Locality [9]). *Let $\mathfrak{A}$ be a model, X a team, $\phi \in \mathrm{FO}(\subseteq)$ a formula, and V a set of variables such that $\mathrm{Fr}(\phi) \subseteq V \subseteq \mathrm{dom}(X)$. Then*

$$\mathfrak{A} \models_X \phi \iff \mathfrak{A} \models_{X \restriction V} \phi.$$

Proposition 5 (Union Closure [9]). *Let $\mathfrak{A}$ be a model, $\mathcal{X}$ a set of teams, and $\phi \in \mathrm{FO}(\subseteq)$ a formula. Then*

$$\forall X \in \mathcal{X} : \mathfrak{A} \models_X \phi \implies \mathfrak{A} \models_{\bigcup \mathcal{X}} \phi.$$

Note that union closure implies the *empty team property*, that is, $\mathfrak{A} \models_\emptyset \phi$ for all inclusion logic formulae ϕ.

The starting point for our investigations is the result by Galliani and Hella [10] characterizing the expressivity of inclusion logic in terms of positive greatest fixed point logic. The latter logic is obtained from greatest fixed-point logic (the dual of least fixed point logic) by restricting to formulae in which fixed point operators occur only positively, that is, within a scope of an even number of negations. In finite models this positive fragment captures the full fixed point logic (with both least and greatest fixed points), and hence it follows from the famous result of Immerman [18] and Vardi [28] that inclusion logic captures polynomial time in finite ordered models.

Theorem 6 ([10]). *Every inclusion logic sentence is equivalent to some positive greatest fixed point logic sentence, and vice versa.*

Theorem 7 ([10]). *A class $\mathcal{C}$ of finite ordered models is in $\mathbf{P}$ iff it can be defined in $\mathrm{FO}(\subseteq)$.*

2.3 Transitive Closure Logic

In Sect. 6 we relate inclusion logic to transitive closure logic, and hence we next give a short introduction to the latter. A $2k$-ary relation R is said to be *transitive* if $(\overline{a}, \overline{b}) \in R$ and $(\overline{b}, \overline{c}) \in R$ imply $(\overline{a}, \overline{c}) \in R$ for k-tuples $\overline{a}, \overline{b}, \overline{c}$. The *transitive closure* of a $2k$-ary relation R, written $\mathrm{TC}(R)$, is defined as the intersection of all $2k$-ary relations $S \supseteq R$ that are transitive. The transitive closure of R can

be alternatively defined as $R_\infty = \bigcup_{i=0}^{\infty} R_i$ for R_i defined recursively as follows: $R_0 = R$ and $R_{i+1} = R \circ R_i$ for $i > 0$; here $A \circ B$ denotes the composition of two relations A and B. Note that $(\overline{a}, \overline{b}) \in R_i$ if and only if there is an R-path of length $i+1$ from $\overline{a}$ to $\overline{b}$.

An assignment s, a model $\mathfrak{A}$, and a formula $\psi(\overline{x}, \overline{y}, \overline{z})$, where $\overline{x}$ and $\overline{y}$ are k-ary, give rise to a $2k$-ary relation defined as follows:

$$R_{\psi,\mathfrak{A},s} = \{\overline{a}\overline{b} \in M^{2k} \mid \mathfrak{A} \models \psi(\overline{a}, \overline{b}, s(\overline{z}))\}.$$

We can now define transitive closure logic. Given a term t, a model $\mathfrak{A}$, and an assignment s, we write $t^{\mathfrak{A},s}$ for the interpretation of t under $\mathfrak{A}, s$, defined in the usual way.

Definition 8 (Transitive Closure Logic). *Transitive closure logic (*TC*) is obtained by extending first-order logic with transitive closure formulae* $[\mathrm{TC}_{\overline{x},\overline{y}}\psi(\overline{x}, \overline{y}, \overline{z})](\overline{t}_0, \overline{t}_1)$ *where* $\overline{t}_0$ *and* $\overline{t}_1$ *are* k*-tuples of terms, and* $\psi(\overline{x}, \overline{y}, \overline{z})$ *is a formula where* $\overline{x}$ *and* $\overline{y}$ *are* k*-tuples of variables. The semantics of the transitive closure formula is defined as follows:*

$$\mathfrak{A} \models_s [\mathrm{TC}_{\overline{x},\overline{y}}\psi(\overline{x}, \overline{y}, \overline{z})](\overline{t}_0, \overline{t}_1) \text{ iff } (\overline{t}_0^{\mathfrak{A},s}, \overline{t}_1^{\mathfrak{A},s}) \in \mathrm{TC}(R_{\psi,\mathfrak{A},s}).$$

Thus, $[\mathrm{TC}_{\overline{x},\overline{y}}\psi(\overline{x}, \overline{y}, \overline{z})](\overline{t}_0, \overline{t}_1)$ is true if and only if there is a ψ-path from $\overline{t}_0$ to $\overline{t}_1$. It is well known that transitive closure logic captures non-deterministic logarithmic space in finite ordered models. In particular, this can be achieved by using only one application of the TC operator. We use below the notation min for the least element of the linear order, and $\overline{\min}$ for the tuple $(\min, \ldots, \min)$. Similarly, $\overline{\max}$ denotes the tuple $(\max, \ldots, \max)$, where max is the greatest element.

Theorem 9 ([19,20]). *A class* $\mathcal{C}$ *of finite ordered models is in* **NL** *iff it can be defined in* TC*. Furthermore, every* TC*-sentence is equivalent in finite ordered models to a sentence of the form*

$$[\mathrm{TC}_{\overline{x},\overline{y}}\alpha(\overline{x}, \overline{y})](\overline{\min}, \overline{\max})$$

where α *is first-order.*

3 Maximal Subteam Membership

In this section we define the maximal subteam membership problem. We first discuss some of its basic properties and then investigate its complexity over quantifier-free inclusion logic formulae.

3.1 Introduction

For a model $\mathfrak{A}$, a team X, and an inclusion logic formula ϕ, we define $\nu(\mathfrak{A}, X, \phi)$ as the unique subteam $Y \subseteq X$ such that $\mathfrak{A} \models_Y \phi$, and $\mathfrak{A} \not\models_Z \phi$ if $Y \subsetneq Z \subseteq X$. Due

to the union closure property $\nu(\mathfrak{A}, X, \phi)$ always exists and it can be alternatively defined as the union of all subteams $Y \subseteq X$ such that $\mathfrak{A} \models_Y \phi$. If ϕ does not contain quantifiers or symbols from the underlying vocabulary, then we may write $\nu(X, \phi)$ instead of $\nu(\mathfrak{A}, X, \phi)$. The maximal subteam membership problem is now given as follows.

Definition 10. *Let $\phi \in \mathrm{FO}(\subseteq)$. Then $\mathsf{MSM}(\phi)$ is the problem of determining whether $s \in \nu(\mathfrak{A}, X, \phi)$ for a given model $\mathfrak{A}$, a team X and an assignment $s \in X$.*

Grädel proved that for any $\mathrm{FO}(\subseteq)$-formula ϕ, there is a formula ψ of positive greatest fixed point logic such that for any model $\mathfrak{A}$ and assignment s, $\mathfrak{A} \models_s \psi$ if and only if s is in the maximal team of $\mathfrak{A}$ satisfying ϕ (see Theorem 24 in [12]). An easy adaptation of the proof shows that $\nu(\mathfrak{A}, X, \phi)$ is also definable in positive greatest fixed point logic. Thus, it follows that every maximal subteam membership problem is polynomial time computable.

Lemma 11. *For every formula $\phi \in \mathrm{FO}(\subseteq)$, $\mathsf{MSM}(\phi)$ is in $\mathbf{P}$.*

In this section we will restrict our attention to maximal subteam problems for quantifier free formulae. Before proceeding to our findings we need to present some auxiliary concepts and results. The following lemmata will be useful below.

Lemma 12. *Let $\alpha, \beta \in \mathrm{FO}(\subseteq)$, and let X be a team of a model $\mathfrak{A}$. Then $\nu(\mathfrak{A}, X, \alpha \vee \beta) = \nu(\mathfrak{A}, X, \alpha) \cup \nu(\mathfrak{A}, X, \beta)$.*

Proof. For "$\subseteq$", note that by definition there are $Y, Z \subseteq X$ such that $Y \cup Z = \nu(\mathfrak{A}, X, \alpha \vee \beta)$, $Y \models \alpha$ and $Z \models \beta$, and hence $Y \subseteq \nu(\mathfrak{A}, X, \alpha)$ and $Z \subseteq \nu(\mathfrak{A}, X, \beta)$. For "$\supseteq$", note that $\nu(\mathfrak{A}, X, \alpha) \cup \nu(\mathfrak{A}, X, \beta)$ satisfies $\alpha \vee \beta$ so it must be contained by $\nu(\mathfrak{A}, X, \alpha \vee \beta)$. □

As an easy corollary we obtain the following lemma.

Lemma 13. *Let $\alpha, \beta \in \mathrm{FO}(\subseteq)$, and assume that $\mathsf{MSM}(\alpha)$ and $\mathsf{MSM}(\beta)$ both belong to a complexity class $C \in \{\mathbf{L}, \mathbf{NL}\}$. Then $\mathsf{MSM}(\alpha \vee \beta)$ is in C.*

The maximal subteam problem for a single inclusion atom $\overline{x} \subseteq \overline{y}$ can be naturally represented using directed graphs. In this representation each assignment forms a vertex, and an assignment s has an outgoing edge to another assignment s' if $s(\overline{x}) = s'(\overline{y})$. Over finite teams an assignment then belongs to the maximal subteam for $\overline{x} \subseteq \overline{y}$ if and only if it is connected to a cycle.[1]

Lemma 14. *Let $\mathfrak{A}$ be a model, X a finite team, $\overline{x}$ and $\overline{y}$ two tuples of the same length from $\mathrm{dom}(X)$, s an assignment of X, and α a first-order formula. Let $G = (X, E)$ be a directed graph where $(s, s') \in E$ iff $s(\overline{x}) = s'(\overline{y})$ and $\mathfrak{A} \models_{\{s,s'\}} \alpha$. Then*

(a) $s \in \nu(\mathfrak{A}, X, \overline{x} \subseteq \overline{y} \wedge \alpha) \iff G$ contains a path from s to a cycle,

[1] We are grateful to Phokion Kolaitis, who pointed out this fact to the second author in a private discussion 2016.

(b) $s \in \nu(\mathfrak{A}, X, \overline{x} \subseteq \overline{y} \wedge \overline{y} \subseteq \overline{x} \wedge \alpha) \iff$ *G contains a path from one cycle to another via s.*

Proof. Assume for the first statement that $s \in \nu(\mathfrak{A}, X, \overline{x} \subseteq \overline{y} \wedge \alpha)$. Then there is a subteam $Y \subseteq X$ such that $s \in Y$ and $\mathfrak{A} \models_Y \overline{x} \subseteq \overline{y} \wedge \alpha$. Thus for each $s' \in Y$ there exists $s'' \in Y$ such that $s''(\overline{y}) = s'(\overline{x})$. Moreover, $\mathfrak{A} \models_{\{s', s''\}} \alpha$, whence $(s', s'') \in E$. In particular there is a non-ending path in G starting from s. Since X is finite, this path necessarily ends in a cycle. Conversely, assume G contains a path from s to a cycle. Then $\mathfrak{A} \models_Y \overline{x} \subseteq \overline{y} \wedge \alpha$ where Y consists of all assignments in the path and cycle. Hence, $s \in \nu(\mathfrak{A}, X, \overline{x} \subseteq \overline{y} \wedge \alpha)$.

For the second statement note that, by the argument above, $s \in \nu(\mathfrak{A}, X, \overline{y} \subseteq \overline{x} \wedge \alpha)$ if and only if $G' = (X, E^{-1})$ contains a path from s to a cycle. But clearly an E^{-1}-path from s to an E^{-1}-cycle is an E-path from an E-cycle to s. □

3.2 Complexity

Next we turn to the computational complexity of maximal subteam membership. In what follows, we give a complete characterization of the maximal subteam problem for arbitrary conjunctions and disjunctions of unary inclusion atoms. A *unary* inclusion atom is an atom of the form $x \subseteq y$ where x and y are single variables. The characterization is given in terms of inclusion graphs.

Definition 15. *Let Σ be a set of unary inclusion atoms over variables in V. Then the* inclusion graph *of Σ is defined as $G_\Sigma = (V, E)$ such that $(x, y) \in E$ iff $x \neq y$ and $x \subseteq y$ appears in Σ.*

We will now prove the following theorem.

Theorem 16. *Let Σ be a finite set of unary inclusion atoms, and let ϕ be the conjunction of all atoms in Σ. Then $\mathsf{MSM}(\phi)$ is*

(a) trivially true if G_Σ has no edges,
(b) **NL***-complete if G_Σ has an edge (x, y) and no other edges except possibly for its inverse (y, x),*
(c) **P***-complete otherwise.*

The first statement above follows from the observation that $\mathsf{MSM}(\phi)$ is true for all inputs if ϕ is a conjunction of trivial inclusion atoms $x \subseteq x$. The second statement is shown by relating to graph reachability. Given a directed graph $G = (V, E)$ and two vertices a and b, the problem REACH is to determine whether G contains a path from a to b. This problem is a well-known complete problem for **NL**.

Lemma 17. $\mathsf{MSM}(x \subseteq y)$ *and* $\mathsf{MSM}(x \subseteq y \wedge y \subseteq x)$ *are* **NL***-complete.*

Proof. **Hardness.** We give a logarithmic space many-one reduction from REACH. Let $G = (V, E)$ be a directed graph, and let $a, b \in V$. W.l.o.g. we can assume that G has no cycles. Note that we obtain a directed acyclic graph

by replacing nodes v with nodes (v, i) and edges (v, v') with edges $((v, i), (v', j))$, for $i, j \in \{1, \ldots, |V|\}$ such that $i < j$. Then $(b, |V|)$ is reachable from $(a, 1)$ in the acyclic graph if and only if b is reachable from a in the initial graph.

Define E' as the extension of E with an extra edge (b, a). Then b is reachable from a in G if and only if a belongs to a cycle in $G' = (V, E')$. We reduce from (G, a, b) to a team $X = \{s_{c,d} \mid (c, d) \in E'\}$ where $s_{u,v}$ maps (y, x) to (u, v) (see Fig. 1). By Lemma 14, b is reachable from a if and only if $s_{b,a} \in \nu(X, \phi)$, where ϕ is either $x \subseteq y$ or $x \subseteq y \wedge y \subseteq x$.

Membership. By Lemma 14 $\mathsf{MSM}(x \subseteq y)$ and $\mathsf{MSM}(x \subseteq y \wedge y \subseteq x)$ reduce to reachability variants that are clearly in **NL**. □

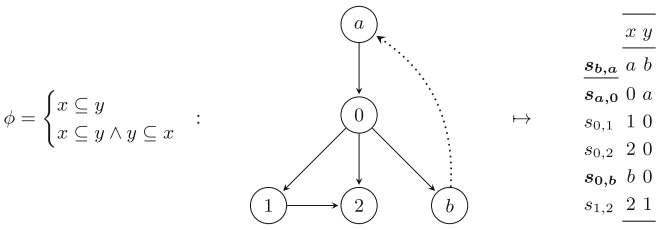


Fig. 1. Reduction from REACH to $\mathsf{MSM}(\phi)$. The input assignment is underlined and the assignments written in bold form a subteam satisfying ϕ.

Next we turn to the third statement of Theorem 16. Recall that membership in **P** follows directly from Lemma 11. For **P**-hardness we reduce from the monotone circuit value problem (see, e.g., [30]). The proof essentially follows from the following lemma.

Lemma 18. $\mathsf{MSM}(x \subseteq z \wedge y \subseteq z)$, $\mathsf{MSM}(x \subseteq y \wedge y \subseteq z)$, *and* $\mathsf{MSM}(x \subseteq y \wedge x \subseteq z)$ *are* **P***-complete.*

Proof. Let ϕ be either $x \subseteq z \wedge y \subseteq z$, $x \subseteq y \wedge y \subseteq z$, or $x \subseteq y \wedge x \subseteq z$. We give a logarithmic-space many-one reduction to $\mathsf{MSM}(\phi)$ from the monotone circuit value problem (MCVP). Given a Boolean word $w \in \{\top, \bot\}^n$, and a Boolean circuit C with n inputs, one output, and gates with labels from {AND,OR}, this problem is to determine whether C outputs $\top$. If C outputs $\top$ on w, we say that it *accepts* w. W.l.o.g. we may assume that the in-degree of each AND and OR gate is 2. We annotate each input node by its corresponding input $\top$ or $\bot$, and each gate by some distinct number $i \in \mathbb{N} \setminus \{0\}$. Then each gate has two child nodes i_L, i_R that are either natural numbers or input values from $\{\top, \bot\}$. Next we construct a team X whose values consist of node annotations $i, \top, \bot$ and distinct copies c_i of AND gates i. The team X is constructed by the following rules (see Fig. 2):

– add $s_0 \colon (x, y, z) \mapsto (1, \top, \top)$ where 1 is the output gate,

– for AND gates i, add $s_{i,0}\colon (x,y,z) \mapsto (i_L, i, c_i)$, $s_{i,1}\colon (x,y,z) \mapsto (i_R, c_i, i)$, and $s_i : (x,y,z) \mapsto (c_i, \top, \top)$,
– for OR gates i, add $s_{i,L}\colon (x,y,z) \mapsto (i_L, i, i)$ and $s_{i,R}\colon (x,y,z) \mapsto (i_R, i, i)$.

We will show that C accepts w iff $s_0 \in \nu(X, \phi)$. For the only-if direction we actually show a slightly stronger claim. That is, we show that $s_0 \in \nu(X, \phi)$ is implied even if ϕ is the conjunction of all unary inclusion atoms between x, y, z.

Assume first that C accepts w. We show how to build a subteam $Y \subseteq X$ such that it includes s_0 and satisfies all unary inclusion atoms between x, y, z. First construct a subcircuit C' of C recursively as follows: add the output gate $\top$ to C'; for each added AND gate i, add both child nodes of i; for each added OR gate i, add a child node of i that is evaluated true under w. In other words, C' is a set of paths from the output gate to the input gates that witnesses the assumption that C accepts w. The team Y will now list the auxiliary values c_i and the gates of C' in each column x, y, z. We construct Y by the following rules:

– add s_0,
– for AND gates i in C', add $s_{i,0}$, $s_{i,1}$, and s_i,
– for OR gates i in C', add $s_{i,P}$ iff i_P is in C', for $P = L, R$.

First observe that Y is formed symmetrically in terms of y and z, and thus these columns share the same values. Consider then the symmetric difference between values in columns x and y. Initially, for $Y = \{s_0\}$, this set is $\{1, \top\}$. An inductive argument now shows that, following the partial ordering induced from C', an application of a construction rule to a gate i of C' modifies the symmetric difference by removing i (and also $\top$ if $\top$ is a child of i) and adding any child node of i that is a gate in C'. In the end the symmetric difference is the empty set, and thus we conclude that Y satisfies all unary inclusion atoms between x, y, z.

Vice versa, consider the standard semantic game between Player I and Player II on the given circuit C and input word w. This game starts from the output gate 1, and at each AND (OR, resp.) gate i Player I (Player II, resp.) selects the next node from its two child nodes i_L and i_R. Player II wins iff the game ends at an input node that is true. By the assumption that $s_0 \in \nu(X, \phi)$ we find a team Y that contains s_0 and satisfies ϕ. Note that Y cannot contain any assignment that maps x to $\bot$. For showing that C accepts w it thus suffices to show that Player II has a strategy which imposes the following restriction: for each visited node annotated by i, we have $s(x) = i$ for some $s \in Y$. At start this holds by the assumption that $s_0 \in Y$. Assume that i is any gate with $s \in Y$ such that $s(x) = i$. If ϕ is $x \subseteq z \wedge y \subseteq z$, we have two cases. If i is an OR gate then we find s' from Y with $s'(y) = s'(z) = i$. Then the strategy of Player II is to select the gate $s'(x)$ as her next step. If i is an AND gate, an application of $x \subseteq z$ gives s' from Y with $s'(z) = i$. Then $s'(y) = c_i$, which means that further application of $y \subseteq z$ yields s'' from Y with $s''(z) = c_i$ and hence $s''(y) = i$. Now $\{s'(x), s''(x)\} = \{i_L, i_R\}$, and thus the claim holds for either selection by Player I. The induction step is analogous for the cases where ϕ is $x \subseteq y \wedge y \subseteq z$ or $x \subseteq y \wedge x \subseteq z$. This concludes the proof. □

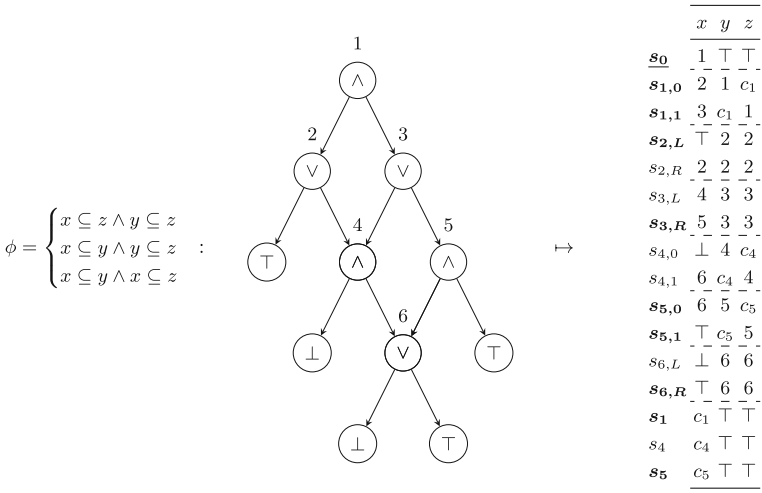


Fig. 2. MCVP and $\mathsf{MSM}(\phi)$

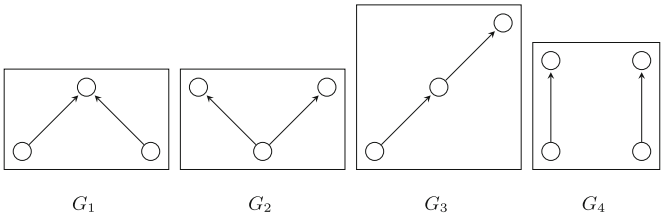


Fig. 3. Subgraphs of G_Σ

The third statement of Theorem 16 now follows. Any G_Σ not covered by the previous lemma has a subgraph of a form depicted in Fig. 3. Of these G_1–G_3 were considered above, and the reduction for G_4 is essentially identical to that for G_1; take a new variable for the new target node and insert values identical to those of z. Additionally, for each node in G_Σ but not in G_i take a copy of any column in the team. That this suffices follows from the arguments of the previous lemmata; in particular, from the fact that any true MCVP instance generates a subteam that satisfies all possible unary inclusion atoms between variables.

Considering disjunctions, observe that MSM over a disjunction of unary inclusion atoms is either trivially true or **NL**-complete. For membership in **NL**, see Lemma 13. For **NL**-hardness of $\mathsf{MSM}(x \subseteq y \vee y \subseteq x)$ we use exactly the same reduction from REACH as in Lemma 17: indeed, by Lemma 12 $s_{b,a} \in \nu(X, x \subseteq y \vee y \subseteq x)$ if and only if $s_{b,a} \in \nu(X, x \subseteq y)$ or $s_{b,a} \in \nu(X, y \subseteq x)$. The first condition holds if and only if a belongs to a cycle in $G' = (V, E')$, which implies that b is reachable from a in G; and the second condition holds if and only if b belongs to a cycle in the graph obtained by inverting the edges of G', which like-

wise implies that b is reachable from a in G. Provided that another non-trivial inclusion atom $u \subseteq v$ appears in the disjunction, then $\{u, v\} \not\subseteq \{x, y\}$ and the values for u, v can be defined in such a way that the maximal subteam for $u \subseteq v$ is empty.

Corollary 19. *Let Σ be a finite set of unary inclusion atoms, and let ϕ be the disjunction of all atoms in Σ. Then $\mathsf{MSM}(\phi)$ is*

(a) trivially true if Σ contains a trivial inclusion atom,
*(b) **NL**-complete otherwise.*

Note that the results of this section generalize to inclusion atoms of higher arity, obtained by replacing variables x with tuples $\overline{x}$ such that all pairs of distinct tuples have no common variables. More complex cases arise if the tuples are allowed to overlap. In the full version of the paper [13] we also consider maximal subteam membership over input teams in which the inclusion atoms reference a key. In such cases the complexity of maximal subteam membership collapses to lower levels. For instance, $\mathsf{MSM}(x \subseteq z)$ ($\mathsf{MSM}(x \subseteq z \wedge y \subseteq z)$, resp.) is **L**-complete (**NL**-complete, resp.) over teams in which z is a key.

4 Consistent Query Answering

The maximal subteam membership problem has a close connection to database repairing which provides a framework for managing inconsistency in databases. An inconsistent database is a database that does not satisfy all the integrity constraints that it is supposed to satisfy. Inconsistency may arise, e.g., from data integration where the task is to bring together data from different sources. Often in practise inconsistency is handled through data cleaning which is the process of identifying and correcting inaccurate data records from databases. An inherent limitation of this approach is its inability to avoid arbitrary choices as consistency can usually be restored in a number of ways. The approach of database repair is to tolerate inconsistencies in databases and investigate reliable answers to queries.

A *database* is an interpretation of a relational vocabulary $\sigma = \{R_1, \ldots, R_n\}$ in which each R_i is associated with an arity $\#R_i$. Given a (finite) set Σ of integrity constraints, a database D is called *inconsistent* (w.r.t. to Σ) if $D \not\models \Sigma$, and *consistent* otherwise. Given a partial order $\leq$ on databases over a fixed σ, and a set Σ of integrity constraints, a *repair* of an inconsistent database I is a database D such that it is consistent and all $D' < D$ are inconsistent. The database D is called a $\oplus$-*repair* if the partial order is defined in terms of symmetric difference: $D \leq D'$ if $D \oplus I \subseteq D' \oplus I$. If additionally D is a subset (superset, resp.) of I, then D is called a *subset-repair* (*superset-repair*, resp.). An *answer* to a first-order query $q = \psi(x_1, \ldots, x_n)$ on a database D is any $(a_1, \ldots, a_n)$ such that D satisfies $\psi(a_1, \ldots, a_n)$, and a *consistent answer* on an inconsistent database I is any value $(a_1, \ldots, a_n)$ such that each repair D of I satisfies $\psi(a_1, \ldots, a_n)$.

Let $* \in \{\oplus$, subset, superset$\}$ and let Σ be a set of integrity constraints. The $*$-*repair checking problem w.r.t.* Σ ($*$-$\mathsf{RC}(\Sigma)$) is to determine, given two

databases D and I, whether D is a $*$-repair of I. Let also q be a Boolean query. The *$*$-consistent query answering problem w.r.t. Σ and q* ($*$-$\mathsf{CQA}(\Sigma, q)$) is to determine, given an inconsistent database I, whether q is true in every $*$-repair of I. *LAV tgds* are first-order formulae of the form

$$\phi = \forall \overline{x}(\psi(\overline{x}) \rightarrow \exists \overline{y}\theta(\overline{x}, \overline{y}))$$

where ψ is a single relational atom and θ is a conjunction of relational atoms, and each variable from $\overline{x}$ occurs in ψ (but not necessarily in θ). *Inclusion dependencies* are the special case of LAV tgds in which also θ is a single relational atom, and no variable occupies two positions in one relational atom. An inclusion dependency is called *unary* if a single variable from $\overline{x}$ appears in exactly one relation position of θ, and it is called *unirelational* if ψ and θ contain the same relation symbol. Note that unary inclusion atoms correspond to unary unirelational inclusion dependencies.

Example 20. Figure 4 depicts a database D consisting of two ternary relations TEACHING and EMPLOYEE. Let Σ consist of a single unary inclusion dependency which states that each `lecturer` in TEACHER is a `name` in EMPLOYEE. The database is inconsistent because Bob is not listed in EMPLOYEE, and it has a unique subset-repair in which (Bob, Mechanics, Spring 2019) is removed from TEACHING. A superset-repair is obtained by adding (Bob, a, b) to EMPLOYEE where a and b are any data values. Such repairs are also $\oplus$-repairs. Consider a query q that returns lecturers located at the Math department. Regardless of the repair type this query has only one consistent answer: Alice.

Consistent query answering and repair checking are known to be tractable for LAV tgds. A *conjunctive query* is a first-order formula of the form $\exists \overline{x}\theta(\overline{x})$ where θ is a conjunction of relational atoms.

Theorem 21 ([25]). *Let $* \in \{\oplus$, subset, superset$\}$, let Σ be a set of LAV tgds, and let q be a conjunctive query. The $*$-repair checking problem w.r.t. Σ and the $*$-consistent query answering problem w.r.t. Σ and q are both solvable in polynomial time.*

TEACHING

lecturer	course	semester
Alice	Analysis	Spring 2019
Alice	Analysis	Fall 2019
Bob	Mechanics	Spring 2019
Carol	Algorithms	Spring 2019

EMPLOYEE

name	department	room
Alice	Math	A321
Carol	CS	B127
Carol	CS	B121

Fig. 4. Database D

Furthermore, it is known that so-called weakly acyclic collections of LAV tgds enjoy subset-repair checking in logarithmic space [1]. Nevertheless, it seems

not much attention in general has been devoted to complexity thresholds within polynomial time. Our results can thus be seen as steps toward this direction as the trichotomy in Theorem 16 extends to repair checking and consistent query answering. Let IC be a collection of finite sets of integrity constraints and let C be a complexity class. We say that the $*$-consistent query answering problem is C-complete for IC if for all $\Sigma \in IC$, $*\text{-}\mathsf{CQA}(\Sigma, q)$ is in C for all Boolean conjunctive queries q and C-complete for some such q.

Theorem 22. *Let $* \in \{\oplus, subset\}$. The subset-repair checking problem and the $*$-consistent query answering problem for finite sets Σ of unary unirelational inclusion dependencies are*

(a) first-order definable if G_Σ has no edges,
*(b) **NL**-complete if G_Σ has an edge (x, y) and no other edges except possibly for its inverse (y, x),*
*(c) **P**-complete otherwise.*

Proof. Since **NL** and **P** are closed under complement, we may consider the complement of subset-repair checking. For the upper bounds note that D is a repair of I if and only if D satisfies Σ (a first-order property) and no tuple in $I \setminus D$ is in the unique subset-repair of I. That Σ has a unique subset-repair follows already from the union closure property of inclusion logic (Proposition 5) (or that of LAV tgds [25]). For the lower bounds note that in our reductions $s \in \nu(X, \phi)$ if and only if $\nu(X, \phi) \neq \emptyset$.[2]

Consider then subset-consistent query answering over a Boolean conjunctive query $q = \exists \overline{x}(R_{i_1}(\overline{x}_1) \wedge \ldots \wedge R_{i_n}(\overline{x}_n))$ where $\overline{x}_1, \ldots, \overline{x}_n$ are subsequences of $\overline{x}$ (note that q may contain multiple relation symbols even though all constraints are unirelational). Considering first the upper bounds, in case (a) q itself may be used for the first-order definition, and in cases (b) and (c) evaluation of the relational atoms $R_{i_1}(\overline{x}_i)$ may be reduced to the maximal subteam membership problem. For the lower bounds we may simply use atomic queries that describe the input assignment for the maximal subteam membership problem. For instance, in case (b) subset-$\mathsf{CQA}(\Sigma, q)$ is **NL**-hard for $q = R(a, b)$ where a and b are constant values from the reduction in Lemma 17. That the result holds also for $\oplus$-consistent query answering follows from the fact that each set of inclusion dependencies Σ has a unique subset repair which is also the unique universal subset repair and the unique universal $\oplus$-repair [25]. A database U is a *universal $*$-repair* of an inconsistent database I if for each conjunctive query q, a tuple is a consistent answer to q on I if and only if it is an answer to q on U and contains only values that appear in I. That is, it only suffices to consult the universal repair for consistent answers. □

[2] In point of fact, the reduction of Lemma 18 requires slight modification: remove assignments $(c_i, \top, \top)$ and add assignments (c_i, j, k) for each assignment $(i, j, k) \in X$ where i is an AND gate.

5 Model Checking

In this section we discuss the model checking problem for quantifier-free inclusion logic formulae. It turns out that the results of the previous section are now easily adaptable. As above, we herein restrict attention to quantifier-free formulae.

Definition 23. *Let* $\phi \in \mathrm{FO}(\subseteq)$. *Then* $\mathsf{MC}(\phi)$ *is the problem of determining whether* $\mathfrak{A} \models_X \phi$, *given a model* $\mathfrak{A}$ *and a team* X.

Hardness results for model checking can now be obtained by relating to maximal subteam membership.

Lemma 24. *Let* $\alpha, \beta \in \mathrm{FO}(\subseteq)$ *be such that*

(i) $\mathsf{Fr}(\alpha) \cap \mathsf{Fr}(\beta) = \emptyset$,
(ii) $\mathsf{MSM}(\alpha)$ *is* C*-hard for* $C \in \{\mathbf{L}, \mathbf{NL}, \mathbf{P}\}$, *and*
(iii) There is a team Y *of* $\mathrm{dom}(\mathfrak{A})$ *with domain* $\mathsf{Fr}(\beta)$ *such that* $\emptyset \neq \nu(\mathfrak{A}, Y, \beta) \subsetneq Y$.

Then $\mathsf{MC}(\alpha \vee \beta)$ *is* C*-hard.*

Proof. Let $(\mathfrak{A}, X, s)$ be an instance of $\mathsf{MSM}(\alpha)$, that is, $\mathfrak{A}$ is a model, X a team over $\mathsf{Fr}(\alpha)$ and $s \in X$. It suffices to define a first-order reduction from $(\mathfrak{A}, X, s)$ to a team X' over $\mathsf{Fr}(\alpha) \cup \mathsf{Fr}(\beta)$ such that $s \in \nu(\mathfrak{A}, X, \alpha)$ iff $\mathfrak{A} \models_{X'} \alpha \vee \beta$. Let $Z_0 := \nu(\mathfrak{A}, Y, \beta)$ and $Z_1 := Y \setminus Z_0$. Note that by condition (i), the union of any $t \in X$ and $t' \in Y$ is an assignment over $\mathsf{Fr}(\alpha) \cup \mathsf{Fr}(\beta)$. We define

$$X' := \{s \cup t' \mid t' \in Z_1\} \cup \{t \cup t' \mid t \in X \setminus \{s\}, t' \in Z_0\}.$$

Since Z_0 and Z_1 are fixed, X' is first-order definable from $(\mathfrak{A}, X, s)$. By Locality (Proposition 4), we have $\nu(\mathfrak{A}, X', \alpha) \upharpoonright \mathsf{Fr}(\alpha) = \nu(\mathfrak{A}, X' \upharpoonright \mathsf{Fr}(\alpha), \alpha) = \nu(\mathfrak{A}, X, \alpha)$, and similarly $\nu(\mathfrak{A}, X', \beta) \upharpoonright \mathsf{Fr}(\beta) = \nu(\mathfrak{A}, Y, \beta) = Z_0$. Hence, it follows from Lemma 12 that $\mathfrak{A} \models_{X'} \alpha \vee \beta$ iff for all $t \cup t' \in X' : t \in \nu(\mathfrak{A}, X, \alpha) \vee t' \in \nu(\mathfrak{A}, Y, \beta)$ iff $s \in \nu(\mathfrak{A}, X, \alpha)$. □

Note that $\mathfrak{A} \models_X \phi$ if and only if $\nu(\mathfrak{A}, X, \phi) = X$ over inclusion logic formulae ϕ. Hence, model checking can be reduced to maximal subteam membership tests over each individual assignment of a team. In particular, this means that model checking is at most as hard as maximal subteam membership; in some cases, as illustrated in Proposition 26(a), it is strictly less hard. Observe that we may omit the case $C = \mathbf{P}$ because $\mathsf{MC}(\alpha)$ is in $\mathbf{P}$ for any $\alpha \in \mathrm{FO}(\subseteq)$ (Theorem 7).

Lemma 25. *Let* $\alpha \in \mathrm{FO}(\subseteq)$ *be such that* $\mathsf{MSM}(\alpha)$ *is in* $C \in \{\mathbf{L}, \mathbf{NL}\}$. *Then* $\mathsf{MC}(\alpha)$ *is in* C.

By Lemmata 13, 24, 25, Theorem 7, and the results of the previous section, the computational complexity of model checking for various quantifier-free inclusion formulae directly follows. The following proposition illustrates some examples. Note that the semantics of the inclusion atom is clearly first-order expressible, and the same applies to any conjunction of inclusion atoms.

Proposition 26.

(a) $\mathsf{MC}(x \subseteq y)$ *and* $\mathsf{MC}(x \subseteq y \wedge u \subseteq v)$ *are first-order definable.*
(b) $\mathsf{MC}(x \subseteq y \vee u \subseteq v)$ *and* $\mathsf{MC}(x \subseteq y \vee u = v)$ *are* **NL***-complete.*
(c) $\mathsf{MC}((x \subseteq z \wedge y \subseteq z) \vee u \subseteq v)$ *and* $\mathsf{MC}((x \subseteq z \wedge y \subseteq z) \vee u = v)$ *are* **P***-complete.*

6 An NL Fragment of Inclusion Logic

Our aim in this section is to find a natural fragment of inclusion logic that captures the complexity class **NL** over ordered finite models. Our approach is to consider preservation of **NL**-computability under the standard logical operators of FO(⊆). By Lemma 13, we already know that **NL**-computability of maximal subteam membership is preserved under disjunctions. However, Theorem 16 shows that conjunction can increase the complexity of the maximal subteam membership problem from **NL** to **P**-complete, and by Proposition 26, combining a conjunction with a disjunction leads to **P**-complete model checking problems. Thus conjunction cannot be used freely in the fragment we aim for.

The following proposition shows that a single universal quantifier can also increase complexity from **NL** to **P**-complete. In the proof we show **P**-hardness by reduction from the **P**-complete problem GAME. An input to GAME is a DAG (directed acyclic graph) $G = (V, E)$ together with a node $a \in V$. Given such input (V, E, a) we consider the following game $\mathsf{Gm}(V, E, a)$ between two players, I and II. During the game the players move a pebble along the edges of G. In the initial position the pebble is on the node $a_0 = a$. If after $2i$ moves the pebble is on a node a_{2i}, then player I chooses a node a_{2i+1} such that $(a_{2i}, a_{2i+1}) \in E$, and player II responds by choosing a node a_{2i+2} such that $(a_{2i+1}, a_{2i+2}) \in E$. The first player unable to move loses the game, and the other player wins it. Since G is a DAG, every play of the game is finite. In particular, the game is determined, i.e., one of the players has a winning strategy. Now we define (V, E, a) to be a positive instance of GAME if and only if player II has a winning strategy in $\mathsf{Gm}(V, E, a)$.

Note that GAME can be seen as a variation of the monotone circuit value problem MCVP. Indeed, it is straighforward to define for a given monotone circuit C and input word w an input (V, E, a) for GAME such that $\mathsf{Gm}(V, E, a)$ simulates the evaluation game of C on w. Thus MCVP is logarithmic-space reducible to GAME. Conversely, it is also easy to give a logarithmic-space reduction from GAME to MCVP.

Proposition 27. *Let* ϕ *be the formula* $\forall z(\neg Eyz \vee z \subseteq x)$. *Then* $\mathsf{MSM}(\phi)$ *is* **P***-complete. Consequently,* $\mathsf{MC}(\phi \vee Euv)$ *is also* **P***-complete.*

Proof. We give now a reduction from GAME to $\mathsf{MSM}(\phi)$. Let (V, E, a) be an input to GAME. Without loss of generality we assume that there is $b \in V$ such that $(b, a) \in E$. Now we simply let $\mathfrak{A} = (V, E)$, $X = \{s : \{x, y\} \to V \mid (s(x), s(y)) \in E\}$ and $s_0 = \{(x, b), (y, a)\}$.

We will use below the notation $I = \{c \in V \mid \forall d \in V : (c, d) \notin E)\}$. Thus, I consists of those elements $c \in V$ for which player II wins $\mathsf{Gm}(V, E, c)$ immediately

because I cannot move. Furthermore, we denote by W the set of all elements $c \in V$ such that player II has a winning strategy in $\mathsf{Gm}(V, E, c)$.

Let Y be the subteam of X consisting of those assignments $s \in X$ for which $s(y) \in W$. We will show that $Y = \nu(\mathfrak{A}, X, \phi)$. Hence in particular $s_0 \in \nu(\mathfrak{A}, X, \phi)$ if and only if (V, E, a) is a positive instance of GAME, as desired.

To prove that $Y \subseteq \nu(\mathfrak{A}, X, \phi)$ it suffices to show that $\mathfrak{A} \models_Y \phi$. Thus let $Z = Y[A/z]$, $Z' = \{s \in Z \mid (s(y), s(z)) \notin E\}$ and $Z'' = (Z \setminus Z') \cup Z_0$, where $Z_0 = \{s \in Z \mid s(z) = s(x) \text{ and } s(y) \in I\}$ (an example of Z'' is illustrated in Fig. 5). Then clearly $\mathfrak{A} \models_{Z'} \neg Eyz$. To show that $\mathfrak{A} \models_{Z''} z \subseteq x$ assume that $s \in Z''$. If $s \in Z \setminus Z'$, then $(s(y), s(z)) \in E$, and since $s \upharpoonright \{x, y\} \in Y$, player II has an answer c to the move $s(z)$ of player I in $\mathsf{Gm}(V, E, s(y))$ such that $c \in W$. Thus, $s^* = \{(x, s(z)), (y, c)\} \in Y$. If $c \in I$, then $s^*(s^*(x)/z) \in Z_0$. Otherwise there is some $d \in V$ such that $(c, d) \in E$, whence $s^*(d/z) \in Z \setminus Z'$. In both cases, there is $s' \in Z''$ such that $s'(x) = s(z)$. Assume then that $s \in Z_0$. Then by the definition of Z_0 we have $s(x) = s(z)$. Thus we see that for every $s \in Z''$ there is $s' \in Z''$ such that $s'(x) = s(z)$. Now we can conclude that $\mathfrak{A} \models_Z \neg Eyz \vee z \subseteq x$, and hence $\mathfrak{A} \models_Y \phi$.

To prove that $\nu(\mathfrak{A}, X, \phi) \subseteq Y$ it suffices to show that if $\mathfrak{A} \models_{Y'} \phi$ for a team $Y' \subseteq X$, then $s(y) \in W$ for every $s \in Y'$. Thus assume that Y' satisfies ϕ and $s \in Y'$. We describe a winning strategy for player II in $\mathsf{Gm}(V, E, s(y))$. If $s(y) \in I$ she has a trivial winning strategy. Otherwise player I is able to move; let $c \in V$ be his first move. Since $\mathfrak{A} \models_{Y'} \phi$, there are $Z', Z'' \subseteq Y'[A/z]$ such that $Y'[A/z] = Z' \cup Z''$, $\mathfrak{A} \models_{Z'} \neg Eyz$ and $\mathfrak{A} \models_{Z''} z \subseteq x$. Consider the assignment $s' = s(c/z) \in Y'[A/z]$. Since $(s'(y), s'(z)) = (s(y), c) \in E$, it must be the case that $s' \in Z''$. Thus there is $s'' \in Z''$ such that $s''(x) = s'(z) = c$. Then the assignment $s^* = s'' \upharpoonright \{x, y\}$ is in $Y' \subseteq X$, whence $(c, d) \in E$, where $d = s^*(y)$. Let d be the answer of player II for the move c of player I. We observe now that using this strategy player II can find a legal answer from the set $\{s^*(y) \mid s^* \in Y'\}$ to any move of player I, as long as player I is able to move. Since the game is determined, this is indeed a winning strategy.

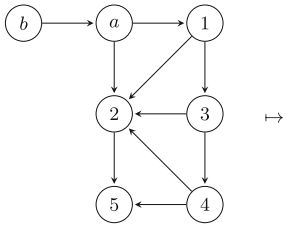
Using Lemma 24, we see that $\mathsf{MC}(\forall z(\neg Eyz \vee z \subseteq x) \vee \beta)$ is **P**-hard for any non-trivial formula β such that $x, y \notin \mathrm{Fr}(\beta)$, in particular for $\beta = Euv$. □

This proposition now demonstrates that, similarly as conjunction, universal quantification cannot be freely used if the goal is to construct a fragment of inclusion logic that captures **NL**. On the positive side, we prove next that existential quantification preserves **NL**-computability. Furthermore, we show that the same holds for conjunction, provided that one of the conjuncts is in FO.

Lemma 28. *Let $\phi \in \mathrm{FO}(\subseteq)$, $\psi \in \mathrm{FO}$, and let X be a team of a model $\mathfrak{A}$. Then*

(a) $\nu(\mathfrak{A}, X, \exists x\phi) = \{s \in X \mid s(a/x) \in X' \text{ for some } a \in A\}$, *where* $X' = \nu(\mathfrak{A}, X[A/x], \phi)$,
(b) $\nu(\mathfrak{A}, X, \phi \wedge \psi) = \nu(\mathfrak{A}, X', \phi)$, *where* $X' = \nu(\mathfrak{A}, X, \psi)$.

Proof. (a) Let $X' = \nu(\mathfrak{A}, X[A/x], \phi)$ and $X'' = \{s \in X \mid s(a/x) \in X' \text{ for some } a \in A\}$. Assume that $Y \subseteq X$ is a team such that $\mathfrak{A} \models_Y \exists x\phi$.



	x	y	z
∘	b	a	$1,2$
	a	1	$2,3$
	a	2	5
	1	2	5
∘	1	3	$2,4$
∘	2	5	2
	3	2	5
	3	4	$2,5$
	4	2	5
∘	4	5	4

Fig. 5. GAME and $\mathsf{MSM}(\forall z(\neg Eyz \vee z \subseteq x))$. The assignments marked by a circle constitute Z''.

Then there is a function $F : X \to \mathcal{P}(A) \setminus \{\emptyset\}$ such that $\mathfrak{A} \models_{Y[F/x]} \phi$, and since clearly $Y[F/x] \subseteq X[A/x]$, we have $Y[F/x] \subseteq X'$. Thus for every $s \in Y$ there is $a \in A$ such that $s(a/x) \in X'$, and hence we see that $Y \subseteq X''$. In particular $\nu(\mathfrak{A}, X, \exists x\phi) \subseteq X''$. To prove the converse inclusion it suffices to show that $\mathfrak{A} \models_{X''} \exists x\phi$. Let $G : X'' \to \mathcal{P}(A) \setminus \{\emptyset\}$ be the function defined by $G(s) = \{a \in A \mid s(a/x) \in X'\}$. By the definition of X'', this function is well-defined and $G(s) \neq \emptyset$ for all $s \in X''$. It is now easy to see that $X''[G/x] = X'$, whence $\mathfrak{A} \models_{X''[G/x]} \phi$, as desired.

(b) Let $X' = \nu(\mathfrak{A}, X, \psi)$ and $X'' = \nu(\mathfrak{A}, X', \psi)$. Assume first that $Y \subseteq X$ is a team such that $\mathfrak{A} \models_Y \phi \wedge \psi$. Then $\mathfrak{A} \models_Y \psi$, whence $Y \subseteq X'$, and furthermore $Y \subseteq X''$, since $\mathfrak{A} \models_Y \phi$. In particular, $\nu(\mathfrak{A}, X, \phi \wedge \psi) \subseteq X''$. On the other hand, by definition $\mathfrak{A} \models_{X''} \phi$. Similarly $\mathfrak{A} \models_{X'} \psi$, whence by downward closure of FO (Corollary 3), $\mathfrak{A} \models_{X''} \psi$. Thus we see that $\mathfrak{A} \models_{X''} \phi \wedge \psi$, which implies that $X'' \subseteq \nu(\mathfrak{A}, X, \phi \wedge \psi)$. □

As a straightforward corollary to this lemma we obtain the following complexity preservation result.

Proposition 29. *Let* $\phi \in \mathrm{FO}(\subseteq)$, $\psi \in \mathrm{FO}$, *and assume that* $\mathsf{MSM}(\phi)$ *is in a complexity class* $C \in \{\mathbf{L}, \mathbf{NL}\}$. *Then*

(a) $\mathsf{MSM}(\exists x\phi)$ *is in* C, *and*
(b) $\mathsf{MSM}(\phi \wedge \psi)$ *is in* C.

Proof. (a) By Lemma 28(a), to check whether a given assignment s is in $\nu(\mathfrak{A}, X, \exists x\phi)$ it suffices to check whether $s(a/x)$ is in $\nu(\mathfrak{A}, X[A/x], \phi)$ for some $a \in A$. Clearly this task can be done in C assuming that $\mathsf{MSM}(\phi)$ is in C.

(b) By Lemma 28(a), it suffices to show that the problem whether an assignment s is in $\nu(\mathfrak{A}, X', \phi)$, where $X' = \nu(\mathfrak{A}, X, \psi)$, can be solved in C with respect to the input $(s, \mathfrak{A}, X)$. Since $\psi \in \mathrm{FO}$, the team X' can be computed in C, whence the claim follows from the assumption that $\mathsf{MSM}(\phi)$ is in C. □

Summarising Lemma 13 and Proposition 29, **NL**-computability of maximal subteam membership is preserved by disjunction, conjunction with first-order

formulas, and existential quantification. Since maximal subteam problem is in **NL** for all first-order formulas and, by Lemma 17, for all inclusion atoms, we define a weak fragment $\mathrm{FO}(\subseteq)_w$ of inclusion logic by the following grammar:

$$\phi ::= \alpha \mid \overline{x} \subseteq \overline{y} \mid \phi \vee \phi \mid \phi \wedge \alpha \mid \exists x \phi,$$

where $\alpha \in \mathrm{FO}$.

Theorem 30. $\mathsf{MC}(\phi)$ *is in* **NL** *for every* $\phi \in \mathrm{FO}(\subseteq)_w$.

Proof. By an easy induction we see that $\mathsf{MSM}(\phi)$ is in **NL** for every $\phi \in \mathrm{FO}(\subseteq)_w$. The claim follows now from Lemma 25.

Vice versa, to show that each **NL** property of ordered models can be expressed in $\mathrm{FO}(\subseteq)_w$, it suffices to show that TC translates to $\mathrm{FO}(\subseteq)_w$ over ordered models.

Theorem 31. *Over finite ordered models,* $\mathrm{TC} \leq \mathrm{FO}(\subseteq)_w$.

Proof. By Theorem 9 we may assume without loss of generality that any TC sentence ϕ is of the form $[\mathrm{TC}_{\overline{x},\overline{y}}\alpha(\overline{x},\overline{y})](\overline{\min},\overline{\max})$ where $\overline{x}$ and $\overline{y}$ are n-ary sequences of variables. We next define an equivalent $\mathrm{FO}(\subseteq)_w$ sentence ϕ'. For two tuples of variables $\overline{x}$ and $\overline{y}$ of the same length, we write $\overline{x} < \overline{y}$ as a shorthand for the formula expressing that $\overline{x}$ is less than $\overline{y}$ in the induced lexicographic ordering, and $\overline{x} = \overline{y}$ for the conjunction expressing that $\overline{x}$ and $\overline{y}$ are pointwise identical. The sentence ϕ' is given as follows:

$$\phi' := \exists \overline{x}\overline{y}\bar{t}_x\bar{t}_y(\psi_1 \wedge \psi_2 \wedge \psi_3 \wedge \psi_4) \tag{1}$$

where

- $\psi_1 := \overline{y}\bar{t}_y \subseteq \overline{x}\bar{t}_x$,
- $\psi_2 := (\bar{t}_x < \overline{\max} \wedge \bar{t}_x < \bar{t}_y \wedge \alpha(\overline{x},\overline{y})) \vee (\bar{t}_x = \overline{\max} \wedge \bar{t}_y = \overline{\min})$,
- $\psi_3 := \neg\bar{t}_x = \overline{\min} \vee \overline{x} = \overline{\min}$, and
- $\psi_4 := \neg\bar{t}_x = \overline{\max} \vee \overline{x} = \overline{\max}$.

Observe that in (1) the tuple $\bar{t}_x$ can be thought of as a counter which indicates an upper bound for the α-path distance of $\overline{x}$ from $\overline{\min}$.

Assuming $\mathfrak{A} \models \phi'$, we find a non-empty team X such that $\mathfrak{A} \models_X \psi_1 \wedge \psi_2 \wedge \psi_3 \wedge \psi_4$. Now, $\mathfrak{A} \models_X \psi_1 \wedge \psi_2$ entails that there is an assignment $s \in X$ mapping $\bar{t}_x$ to $\overline{\min}$, and $\mathfrak{A} \models_X \psi_3$ implies that s maps $\overline{x}$ to $\overline{\min}$, too. Then $\mathfrak{A} \models_X \psi_1 \wedge \psi_2$ entails that there is an α-path from $\overline{\min}$ to $s'(\overline{x})$ for some $s' \in X$ with $s'(\bar{t}_x) = \overline{\max}$. Lastly, by $\mathfrak{A} \models_X \psi_4$ it follows that $s'(\overline{x}) = \overline{\max}$ which shows that $[\mathrm{TC}_{\overline{x},\overline{y}}\alpha(\overline{x},\overline{y})](\overline{\min},\overline{\max})$.

Assume then that $[\mathrm{TC}_{\overline{x},\overline{y}}\alpha(\overline{x},\overline{y})](\overline{\min},\overline{\max})$, that is, there is an α-path $\overline{v}_1,\ldots,\overline{v}_k$ where $\overline{v}_1 = \overline{\min}$ and $\overline{v}_k = \overline{\max}$. We may assume that there are no cycles in the path. Let $\overline{a}_i$ denote the ith element in the lexicographic ordering of A^n. We let $X = \{s_1,\ldots,s_k\}$ be such that $(\overline{x},\overline{y},\bar{t}_x,\bar{t}_y)$ is mapped to $(\overline{v}_i,\overline{v}_{i+1},\overline{a}_i,\overline{a}_{i+1})$ by s_i, for $i = 1,\ldots,k-2$, to $(\overline{v}_{k-1},\overline{v}_k,\overline{a}_{k-1},\overline{\max})$ by s_{k-1}, and to $(\overline{v}_k,\overline{v}_1,\overline{\max},\overline{\min})$ by s_k. It is straightforward to verify that $\mathfrak{A} \models_X \psi_1 \wedge \psi_2 \wedge \psi_3 \wedge \psi_4$ from which it follows that $\mathfrak{A} \models \phi'$. ◻

It now follows by the above two theorems and Theorem 9 that $\mathrm{FO}(\subseteq)_w$ captures **NL**.

Theorem 32. *A class $\mathcal{C}$ of finite ordered models is in* **NL** *iff it can be defined in* $\mathrm{FO}(\subseteq)_w$.

7 Conclusion

We have studied the complexity of inclusion logic from the vantage point of two computational problems: the maximal subteam membership and the model checking problems for fixed inclusion logic formulae. We gave a complete characterization for the former in terms of arbitrary conjunctions/disjunctions of unary inclusion atoms. In particular, we showed that maximal subteam membership is **P**-complete for any conjunction of unary inclusion atoms, provided that the conjunction contains two non-trivial atoms that are not inverses of each other. Using these results we characterized the complexity of model checking for several quantifier-free inclusion logic formulae. We leave it for future research to address the complexity of quantifier-free inclusion logic formulae that involve inclusion atoms of higher arity and both disjunctions and conjunctions.

Assuming the presence of quantifiers we presented a simple universally quantified formula that has **P**-complete maximal subteam membership problem. Finally, we defined a fragment of inclusion logic, obtained by restricting the scope of conjunction and universal quantification, that captures non-deterministic logarithmic space over finite ordered models.

Acknowledgements. We are grateful to Phokion Kolaitis, who raised the questions on the complexity of quantifier-free formulas of inclusion logic in a private discussion with the second author in 2016.

References

1. Afrati, F.N., Kolaitis, P.G.: Repair checking in inconsistent databases: algorithms and complexity. In: 12th International Conference on Database Theory - ICDT 2009, St. Petersburg, Russia, 23–25 March 2009, pp. 31–41 (2009)
2. Arenas, M., Bertossi, L.E., Chomicki, J.: Consistent query answers in inconsistent databases. In: Proceedings of the Eighteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, Philadelphia, Pennsylvania, USA, 31 May–2 June 1999, pp. 68–79 (1999)
3. Chomicki, J., Marcinkowski, J.: Minimal-change integrity maintenance using tuple deletions. Inf. Comput. **197**(1–2), 90–121 (2005)
4. Corander, J., Hyttinen, A., Kontinen, J., Pensar, J., Väänänen, J.: A logical approach to context-specific independence. In: Väänänen, J., Hirvonen, Å., de Queiroz, R. (eds.) WoLLIC 2016. LNCS, vol. 9803, pp. 165–182. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-52921-8_11
5. Durand, A., Hannula, M., Kontinen, J., Meier, A., Virtema, J.: Approximation and dependence via multiteam semantics. Ann. Math. Artif. Intell. **83**, 297–320 (2018)

6. Durand, A., Kontinen, J., de Rugy-Altherre, N., Väänänen, J.: Tractability frontier of data complexity in team semantics. In: Proceedings Sixth International Symposium on Games, Automata, Logics and Formal Verification, GandALF 2015, Genoa, Italy, 21–22nd September 2015, pp. 73–85 (2015)
7. Ebbing, J., Hella, L., Meier, A., Müller, J.-S., Virtema, J., Vollmer, H.: Extended modal dependence logic $\mathcal{EMDL}$. In: Libkin, L., Kohlenbach, U., de Queiroz, R. (eds.) WoLLIC 2013. LNCS, vol. 8071, pp. 126–137. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-39992-3_13
8. Ebbing, J., Kontinen, J., Müller, J.-S., Vollmer, H.: A fragment of dependence logic capturing polynomial time. Log. Methods Comput. Sci. **10**(3) (2014)
9. Galliani, P.: Inclusion and exclusion dependencies in team semantics: on some logics of imperfect information. Ann. Pure Appl. Log. **163**(1), 68–84 (2012)
10. Galliani, P., Hella, L.: Inclusion logic and fixed point logic. In: Rocca, S.R.D. (ed.) Computer Science Logic 2013 (CSL 2013). Leibniz International Proceedings in Informatics (LIPIcs), Dagstuhl, Germany, vol. 23, pp. 281–295. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik (2013)
11. Grädel, E.: Model-checking games for logics of imperfect information. Theor. Comput. Sci. **493**, 2–14 (2012)
12. Grädel, E.: Games for inclusion logic and fixed-point logic. In: Abramsky, S., Kontinen, J., Väänänen, J., Vollmer, H. (eds.) Dependence Logic, pp. 73–98. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-31803-5_5
13. Hannula, M., Hella, L.: Complexity thresholds in inclusion logic. CoRR, abs/1903.10706 (2019)
14. Hannula, M., Kontinen, J.: A finite axiomatization of conditional independence and inclusion dependencies. Inf. Comput. **249**, 121–137 (2016)
15. Hannula, M., Kontinen, J., Virtema, J.: Polyteam semantics. In: Artemov, S., Nerode, A. (eds.) LFCS 2018. LNCS, vol. 10703, pp. 190–210. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-72056-2_12
16. Hannula, M., Kontinen, J., Virtema, J., Vollmer, H.: Complexity of propositional logics in team semantic. ACM Trans. Comput. Log. **19**(1), 2:1–2:14 (2018)
17. Hodges, W.: Compositional semantics for a language of imperfect information. J. Interest Group Pure Appl. Log. **5**(4), 539–563 (1997)
18. Immerman, N.: Relational queries computable in polynomial time. Inf. Control **68**(1), 86–104 (1986)
19. Immerman, N.: Languages that capture complexity classes. SIAM J. Comput. **16**(4), 760–778 (1987)
20. Immerman, N.: Nondeterministic space is closed under complementation. SIAM J. Comput. **17**(5), 935–938 (1988)
21. Kontinen, J.: Coherence and computational complexity of quantifier-free dependence logic formulas. Studia Logica **101**(2), 267–291 (2013)
22. Kontinen, J., Kuusisto, A., Lohmann, P., Virtema, J.: Complexity of two-variable dependence logic and if-logic. Inf. Comput. **239**, 237–253 (2014)
23. Koutris, P., Wijsen, J.: Consistent query answering for primary keys in logspace. In: 22nd International Conference on Database Theory, ICDT 2019, Lisbon, Portugal, 26–28 March 2019, pp. 23:1–23:19 (2019)
24. Pacuit, E., Yang, F.: Dependence and independence in social choice: Arrow's theorem. In: Abramsky, S., Kontinen, J., Väänänen, J., Vollmer, H. (eds.) Dependence Logic, pp. 235–260. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-31803-5_11
25. ten Cate, B., Fontaine, G., Kolaitis, P.G.: On the data complexity of consistent query answering. Theory Comput. Syst. **57**(4), 843–891 (2015)

26. Väänänen, J.: Dependence Logic. Cambridge University Press, Cambridge (2007)
27. Väänänen, J.: Modal dependence logic. In: Apt, K.R., van Rooij, R. (eds.) New Perspectives on Games and Interaction. Amsterdam University Press, Amsterdam (2008)
28. Vardi, M.Y.: The complexity of relational query languages. In: Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing, pp. 137–146. ACM (1982)
29. Virtema, J.: Complexity of validity for propositional dependence logics. Inf. Comput. **253**, 224–236 (2017)
30. Vollmer, H.: Introduction to Circuit Complexity - A Uniform Approach. EATCS Series. Springer, Heidelberg (1999). https://doi.org/10.1007/978-3-662-03927-4
31. Yang, F., Väänänen, J.: Propositional logics of dependence. Ann. Pure Appl. Logic **167**(7), 557–589 (2016)