

Language, Information, and Computation : 26th International Workshop, WoLLIC
2019, Utrecht, The Netherlands, July 2-5, 2019, Proceedings. 2019, 301-322.
https://doi.org/10.1007/978-3-662-59533-6_19

Complexity Thresholds in Inclusion Logic

Miika Hannula¹ and Lauri Hella²

¹University of Helsinki, Finland

²Tampere University, Finland

March 27, 2019

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.

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 [25]. Dependence logic is a logical formalism that extends first-order logic with novel atomic formulae $\text{dep}(x_1, \dots, x_n)$ expressing that a variable x_n depends on variables x_1, \dots, 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 [15, 28, 30], modal logic [7, 26], probabilistic logics [5], and two-variable logics [21]. Recent research has also pursued connections and applications of dependence logic to fields such as database theory [13, 14], Bayesian networks [4], and social choice theory [23]. A common notion underlying all these endeavours is that of team semantics. Team semantics, introduced by Hodges in [16], 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 [25, 5, 14].

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 \bar{x} and \bar{y} having same length, an inclusion atom $\bar{x} \subseteq \bar{y}$ expresses that the set of values of \bar{x} is included in the set of values of \bar{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 [25], and thus to non-deterministic polynomial time (NP), this finding established inclusion logic as the first team-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 $\text{MSM}(\phi)$ 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 [24].

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, 22, 24]. 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 $\text{MSM}(\phi)$ just in case it is in the unique subset repair of I . Note however, that in $\text{MSM}(\phi)$, 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, 20] 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 [20]. The results of this paper, in contrast, demonstrate that the complexity of inclusion logic formulae is particularly sensitive to conjunctions. We show that $\text{MSM}(\phi)$ 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 $\text{MSM}(\phi)$ 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 $\phi \vee x \subseteq y$ is **NL**-hard, while that of $\phi \vee (x \subseteq z \wedge y \subseteq z)$ is **P**-complete.

We also present a safety game for the maximal subteam membership problem. Using this game we examine instances of the maximal subteam membership problem in which the inclusion atoms refer to a key, that is, all inclusion atoms are of the form $x \subseteq y$ where y is a variable which uniquely determines all the remaining variables. We give example formulae for which the thresholds between **NL** and **P** drop down to **L** and **NL** under these assumptions.

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 [18, 19].

2 Preliminaries

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

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

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 $\text{Fr}(\phi)$ 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 $s = (a_1, \dots, a_n)$ if the underlying ordering (x_1, \dots, x_n) of the domain is clear from the context. A *team* X of A with domain $\text{dom}(X) = \{x_1, \dots, x_n\}$ is a set of assignments from $\text{dom}(X)$ into A . For $V \subseteq \text{dom}(X)$, the *restriction* $X \upharpoonright V$ of a team X is defined as $\{s \upharpoonright V \mid s \in X\}$. If X is a team, $V \subseteq \text{dom}(X)$, and $F : X \rightarrow \mathcal{P}(A) \setminus \{\emptyset\}$, 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 $F : X \rightarrow \mathcal{P}(A) \setminus \{\emptyset\}$,
- $\mathfrak{A} \models_X \forall x\phi$ 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 any 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 [25]). *For all models \mathfrak{A} , teams X , and formulae $\phi \in \text{FO}$,*

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

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

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

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

2.2 Inclusion Logic

Inclusion logic ($\text{FO}(\subseteq)$) is defined as the extension of FO by inclusion atoms.

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

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

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, $\phi \in \text{FO}(\subseteq)$ a formula, and V a set of variables such that $\text{Fr}(\phi) \subseteq V \subseteq \text{dom}(X)$. Then*

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

Proposition 5 (Union Closure [9]). *Let \mathfrak{A} be a model, \mathcal{X} a set of teams, and $\phi \in \text{FO}(\subseteq)$ 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 [17] and Vardi [27] 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 $\text{FO}(\subseteq)$.*

2.3 Transitive Closure Logic

In Section 6 we will explore connections between inclusion logic and 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 $(\bar{a}, \bar{b}) \in R$ and $(\bar{b}, \bar{c}) \in R$ imply $(\bar{a}, \bar{c}) \in R$ for k -tuples $\bar{a}, \bar{b}, \bar{c}$. The *transitive closure* of a $2k$ -ary relation R , written $\text{TC}(R)$, 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 $(\bar{a}, \bar{b}) \in R_i$ iff there is an R -path of length $i + 1$ from \bar{a} to \bar{b} .

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

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

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 $[\text{TC}_{\bar{x}, \bar{y}} \psi(\bar{x}, \bar{y}, \bar{z})](\bar{t}_0, \bar{t}_1)$ where \bar{t}_0 and \bar{t}_1 are k -tuples of terms, and $\psi(\bar{x}, \bar{y}, \bar{z})$ is a formula where \bar{x} and \bar{y} are k -tuples of variables. The semantics of the transitive closure formula is defined as follows:

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

Thus, $[\text{TC}_{\bar{x}, \bar{y}} \psi(\bar{x}, \bar{y}, \bar{z})](\bar{t}_0, \bar{t}_1)$ is true if and only if there is a ψ -path from \bar{t}_0 to \bar{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, \dots, \min) . Similarly, $\overline{\max}$ denotes the tuple (\max, \dots, \max) , where \max is the greatest element.

Theorem 9 ([18, 19]). *A class \mathcal{C} of finite ordered models is in \mathbf{NL} iff it can be defined in TC. Furthermore, every TC-sentence is equivalent in finite ordered models to a sentence of the form*

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

where α is first-order.

3 Maximal Subteam Membership

In this section we define the maximal subteam membership problem and discuss some of its basic properties. We also define a safety game for quantifier-free inclusion logic formulae. This game will be used later to facilitate some proofs regarding the complexity of the maximal subteam membership problem.

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 any 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 $\phi \in \text{FO}(\subseteq)$. Then $\text{MSM}(\phi)$ 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 $\text{FO}(\subseteq)$ -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 $\phi \in \text{FO}(\subseteq)$, $\text{MSM}(\phi)$ 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 $\alpha, \beta \in \text{FO}(\subseteq)$, 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 $\nu(\mathfrak{A}, X, \alpha \vee \beta)$. \square

As an easy corollary we obtain the following lemma.

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

The maximal subteam problem for a single inclusion atom $\bar{x} \subseteq \bar{y}$ 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 $s(\bar{x}) = s'(\bar{y})$. Over finite teams an assignment then belongs to the maximal subteam for $\bar{x} \subseteq \bar{y}$ if and only if it is connected to a cycle.¹

Lemma 14. Let \mathfrak{A} be a model, X a finite team, \bar{x} and \bar{y} two tuples of the same length from $\text{dom}(X)$, 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 $s(\bar{x}) = s'(\bar{y})$ and $\mathfrak{A} \models_{\{s, s'\}} \alpha$. Then

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

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

Proof. Assume for the first statement that $s \in \nu(\mathfrak{A}, X, \bar{x} \subseteq \bar{y} \wedge \alpha)$. Then there is a subteam $Y \subseteq X$ such that $s \in Y$ and $\mathfrak{A} \models_Y \bar{x} \subseteq \bar{y} \wedge \alpha$. Thus for each $s' \in Y$ there exists $s'' \in Y$ such that $s'' = s'(\bar{x})$. 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 $\mathfrak{A} \models_Y \bar{x} \subseteq \bar{y} \wedge \alpha$ where Y consists of all assignments in the path and cycle. Hence, $s \in \nu(\mathfrak{A}, X, \bar{x} \subseteq \bar{y} \wedge \alpha)$.

For second statement note that, by the argument above, $s \in \nu(\mathfrak{A}, X, \bar{y} \subseteq \bar{x} \wedge \alpha)$ 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 . \square

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

3.2 Safety Game

In this section we present a version of a safety game for the maximal subteam problem of inclusion logic. Our presentation is also related to the safety games for inclusion logic examined in [12]. We present a safety game for a quadruple $(\mathfrak{A}, X, s, \phi)$, written $\text{safety}(\mathfrak{A}, X, s, \phi)$, where s is an assignment of a team X , and ϕ is a quantifier-free formula. The main result of the section shows that the maximal subteam problem $\text{MSM}(\phi)$ over X and s can be characterized in terms of this game.

We assume that the reader is familiar with basic terminology on trees. We associate each quantifier-free $\phi \in \text{FO}(\subseteq)$ with a labeled rooted tree T_ϕ such that the root of the tree is labeled by ϕ and each node labeled by $\psi_0 \vee \psi_1$ or $\psi_0 \wedge \psi_1$ has two children labeled by ψ_0 and ψ_1 . Notice that two different nodes may have the same label. The safety game for $(\mathfrak{A}, X, s, \phi)$ can now be interpreted as a pebble game where assignments of a team X form a stack of pebbles of which one at a time is placed on a node of T_ϕ . Legal moves of the game then consist of moving the pebble up and down through the tree, removing the pebble from a leaf, and placing a new pebble on a leaf. The starting position is to have s placed on the root, and the winning condition for Player I is to arrive at a position where the game terminates. If no such position is ever reached, Player II wins.

Definition 15 (Safety Game). Let $\phi \in \text{FO}(\subseteq)$ be quantifier-free, and let s_0 be an assignment in a team X of a model \mathfrak{A} . The *safety game* $\text{safety}(\mathfrak{A}, X, s_0, \phi)$ has two players I and II, and the game moves consist of *positions* (s, n) and (n, s) where $s \in X$ and n is a node of T_ϕ . The game starts with the position (s_0, r) , where r is the root, and given a position (s, n) , the game proceeds as follows:

- (i) If n is labeled by a conjunction, then Player I selects a position (s, n') where n' is a child of n .
- (ii) If n is labeled by a disjunction, then Player II selects a position (s, n') where n' is a child of n .
- (iii) If n is labeled by a literal ψ , then the game ends if $\mathfrak{A} \not\models_s \psi$. Otherwise, Player I selects a position (s, n') such that n is a descendant of n' .
- (iv) If n is labeled by $\bar{x} \subseteq \bar{y}$, then the game ends if there is no $s' \in X$ such that $s(\bar{x}) = s'(\bar{y})$. Otherwise, Player I either
 - selects a position (s, n') such that n is a descendant of n' , or
 - selects the position (n, s) .

Given a position (n, s) , the game proceeds as follows:

- (v) Player II selects a position (s', n) such that $s(\bar{x}) = s'(\bar{y})$.

Player I wins if the game ends after a finite number of moves by the players. Otherwise, Player II wins.

A strategy for a Player is a mapping π on positions such that

- $\pi((s, n)) \in \{(s, n') \mid n' \text{ is a child of } n\}$, for a non-leaf n ,
- $\pi((s, n)) \in \{(s, n') \mid n \text{ is a descendant of } n'\}$, for a leaf n labeled by a literal,
- $\pi((s, n)) \in \{(s, n') \mid n \text{ is a descendant of } n'\} \cup \{(n, s)\}$, for a leaf n labeled by $\bar{x} \subseteq \bar{y}$.
- $\pi((n, s)) \in \{(s', n) \mid s' \in X, s(\bar{x}) = s'(\bar{y})\}$, for a leaf n labeled by $\bar{x} \subseteq \bar{y}$.

Player $A \in \{I, II\}$ has a winning strategy for $\text{safety}(\mathfrak{A}, X, s_0, \phi)$ if there is a strategy π_A such that A wins every game that she plays according to π_A . That is, A wins any game where she selects the position $\pi_A(p)$ on her moves on p .

Note that if ϕ does not contain any symbols from the underlying vocabulary, the outcome of $\text{safety}(\mathfrak{A}, X, s, \phi)$ is independent of \mathfrak{A} , and thus we write $\text{safety}(X, s, \phi)$ instead.

Next we show that the safety game above gives rise to a characterization of the maximal subteam problem.

Theorem 16. *Let $\phi \in \text{FO}(\subseteq)$ be quantifier-free, and let s be an assignment in a team X of a model \mathfrak{A} . Then $s \in \nu(\mathfrak{A}, X, \phi)$ iff Player II has a winning strategy in $\text{safety}(\mathfrak{A}, X, s, \phi)$.*

Proof. For the “only-if” direction, we define top-down recursively for each node $n \in T_\phi$ a team X_n such that

- $X_r := \nu(\mathfrak{A}, X, \phi)$ for the root r ,
- $X_n := \nu(\mathfrak{A}, X_{n'}, \psi)$, for a child n of a node n' where n is labeled by ψ .

It follows that $X_n \models \psi$ for n with label ψ ; $X_n = X_{n_0} = X_{n_1}$ for conjunction-labeled n with children n_0, n_1 ; and $X_n = X_{n_0} \cup X_{n_1}$ for disjunction-labeled n with children n_0, n_1 . The strategy of Player II is now the following. If n is labeled by disjunction, then (s, n) is mapped to some (s, n_i) where n_i is a child of n such that $s \in X_{n_i}$, and if n is labeled by $\bar{x} \subseteq \bar{y}$, then (n, s) is mapped to some (s', n) such that $s(\bar{x}) = s'(\bar{y})$ and $s' \in X_n$. We leave it to the reader to check that this strategy is well-defined and winning.

For the “if” direction, assume Player II has a winning strategy π . For a node n of T_ϕ , we let X_n be the set of all assignments $s \in X$ for which there exists a game where Player II plays according to her winning strategy and position (s, n) is played at some point of the game. Consider any assignment s from X_n for a node n labeled by $\bar{x} \subseteq \bar{y}$. This means there is a game where position (s, n) is played, and thus also a game where (n, s) , and furthermore $\pi((n, s)) = (n, s')$ is played. Consequently, an assignment $s' \in X_n$ exists such that $s(\bar{x}) = s'(\bar{y})$. By analogous reasoning we obtain that $X_n \models \psi$ for all other types of nodes n with label ψ , too. Furthermore, $X_n = X_{n_0} = X_{n_1}$ for conjunction-labeled n with children n_0, n_1 , and $X_n = X_{n_0} \cup X_{n_1}$ for disjunction-labeled n with children n_0, n_1 . In particular, $X_r \models \phi$ and $s \in X_r$, and hence $s \in \nu(X, \phi)$. \square

Given that X is finite, it makes sense to consider bounded length restrictions of the safety game. We let $\text{safety}_k(\mathfrak{A}, X, s, \phi)$ denote the version of $\text{safety}(\mathfrak{A}, X, s, \phi)$ in which, starting position (s, r) excluded, positions of the form (s, n) , i.e., pairs whose left element is an assignment and right element a node, are played at most k times. Player I wins $\text{safety}_k(\mathfrak{A}, X, s, \phi)$ if the game terminates before such assignment-node pairs have been played k times. Otherwise, if exactly k plays of such nodes appear, Player II wins. The next lemma will be useful later.

Lemma 17. *Let $\phi \in \text{FO}(\subseteq)$ be quantifier-free and such that T_ϕ has k nodes, and let s be an assignment of a team X that is of size l . Then Player II has a winning strategy for $\text{safety}(\mathfrak{A}, X, s, \phi)$ iff she has a winning strategy for $\text{safety}_{k \cdot l}(\mathfrak{A}, X, s, \phi)$.*

Proof. By the end of $\text{safety}_{k \cdot l}(\mathfrak{A}, X, s, \phi)$, positions of the form (s, n) , the root position included, have occurred $k \cdot l + 1$ many times, i.e., some position (s, n) has occurred twice. Every time such a repetition is encountered, we may assume that we continue the game from the first occurrence of (s, n) . Since the strategy of Player II is safe for $k \cdot l$ assignment-node moves, we conclude that $\text{safety}(\mathfrak{A}, X, s, \phi)$ never terminates. Hence, Player II wins. \square

4 Complexity of Maximal Subteam Membership

Next we examine the computational complexity of maximal subteam membership. First in Section 4.1 we investigate this problem over arbitrary teams, and then in Section 4.1 we restrict attention to inputs in which the inclusion atoms refer to a key. In Section 4.3 we discuss the implications of our results to consistent query answering.

4.1 Arbitrary Teams

We give a complete characterization of the maximal subteam problem for arbitrary conjunctions 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 18. 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 19. *Let Σ be a finite set of unary inclusion atoms, and let ϕ be the conjunction of all atoms in Σ . Then $\text{MSM}(\phi)$ is*

- (a) *trivially true if G_Σ has no edges,*

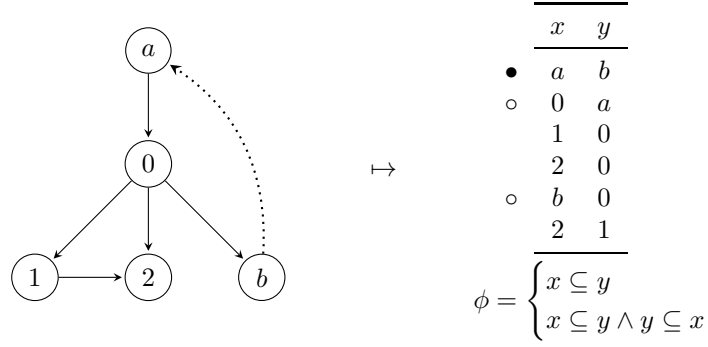


Figure 1: Reduction from REACH to $\text{MSM}(\phi)$. The black circle marks the input assignment and all the circles together mark a subteam satisfying ϕ .

- (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 $\text{MSM}(\phi)$ 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**, and it will also be applied later in Section 4.2 where the complexity of $\text{MSM}(x \subseteq y \wedge u \subseteq v)$ over teams with keys y and v is examined.

Lemma 20. $\text{MSM}(x \subseteq y)$ and $\text{MSM}(x \subseteq y \wedge y \subseteq x)$ are **NL**-complete.

Proof. Hardness. We give a logarithmic space many-one reduction from REACH. Let $G = (V, E)$ be a graph, and let $a, b \in V$. W.l.o.g. we can assume G has no cycles. 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 $X = \{s_{d,c} \mid (c, d) \in E'\}$ where $s_{u,v}$ maps (x, y) to (u, v) (see Fig. 1). By Lemma 14, b is reachable from a if and only if $s_{a,b} \in \nu(X, \phi)$, where ϕ is either $x \subseteq y$ or $x \subseteq y \wedge y \subseteq x$.

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

Next we turn to the third statement of Theorem 19. Recall that membership in **P** follows directly from Lemma 11. For **P**-hardness we reduce from the monotone circuit value problem (see, e.g., [29]). The proof essentially follows from the following lemma.

Lemma 21. $\text{MSM}(x \subseteq z \wedge y \subseteq z)$, $\text{MSM}(x \subseteq y \wedge y \subseteq z)$, and $\text{MSM}(x \subseteq y \wedge x \subseteq z)$ 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 $\text{MSM}(\phi)$ from the monotone circuit value problem (MCVP). Given a Boolean word $w \in \{\top, \perp\}^n$, and a Boolean circuit C with n inputs, one output, and gates with labels from $\{\text{AND}, \text{OR}\}$, 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 \perp , and each gate by some distinct number $i \in \mathbb{N}$. Then each gate has two child nodes i_L, i_R that are either natural numbers or input values from $\{\top, \perp\}$. Next we construct a team X whose values consist of node annotations i, \top, \perp and distinct copies c_i of AND gates i . The team X is constructed by the following rules (see Fig. 2):

- add $s_0: (x, y, z) \mapsto (1, \top, \top)$ where 1 is the output gate,
- for AND gates i , add $s_{i,0}: (x, y, z) \mapsto (i_L, i, c_i)$, $s_{i,1}: (x, y, z) \mapsto (i_R, c_i, i)$, and $s_i: (x, y, z) \mapsto (c_i, \top, \top)$,
- for OR gates i , add $s_{i,L}: (x, y, z) \mapsto (i_L, i, i)$ and $s_{i,R}: (x, y, z) \mapsto (i_R, i, i)$.

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 $P = L, R$.

It follows from the construction 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 \perp . 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. \square

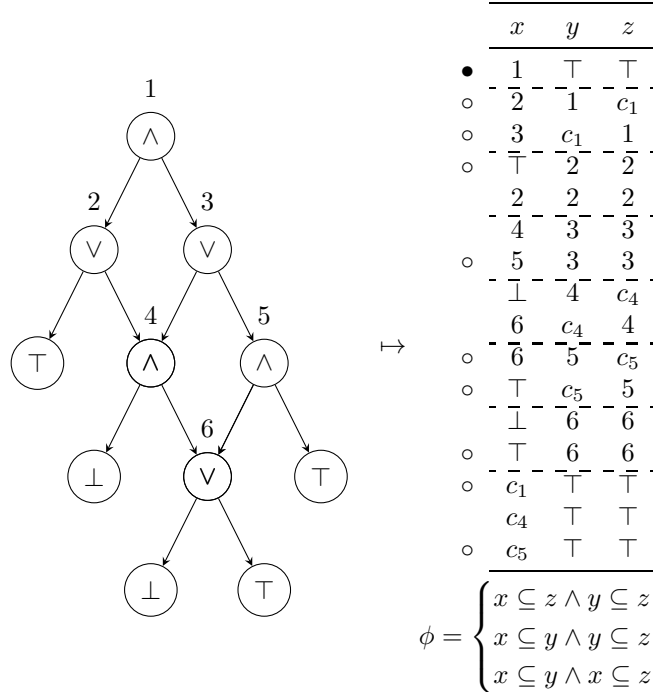


Figure 2: MCVP and $\text{MSM}(\phi)$

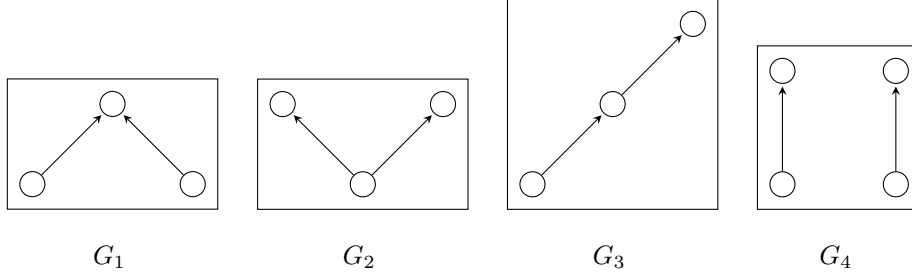


Figure 3: Subgraphs of G_Σ

The third statement of Theorem 19 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 one may use the reduction of Lemma 20 for $\text{MSM}(x \subseteq y \vee y \subseteq x)$. 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 22. *Let Σ be a finite set of unary inclusion atoms, and let ϕ be the disjunction of all atoms in Σ . Then $\text{MSM}(\phi)$ 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 \bar{x} such that all pairs of distinct tuples have no common variables. More complex cases arise if the tuples are allowed to overlap.

4.2 Teams with a Key

In relational database management inclusion atoms usually appear in form of a foreign key that has the purpose of securing referential integrity upon databases. In this section we investigate the maximal subteam problems in a framework where inclusion atoms correspond to uni-relational foreign keys. That is, we consider inclusion atoms $x \subseteq y$ over teams on which the variable y is a key. Given this additional restriction, we observe that the maximal subteam and model checking problems both collapse to lower computational levels. The complexity of the maximal subteam problem for $x \subseteq y$ collapses from **NL** to **L**, and for $x \subseteq z \wedge y \subseteq z$ it collapses from **P** to **NL**.

Definition 23. Let X be a team over a set of variables V , and let U be a subset of V . Then U is a *key* on X if for all $s, s' \in X : s(U) = s'(U) \implies s = s'$.

First we show that maximal subteam for single inclusion atom is **L**-complete if the inclusion atom refers to a key.

Theorem 24. *$\text{MSM}(x \subseteq y)$ over teams for which y is a key is **L**-complete.*

Proof. Hardness. We show a first-order many-one reduction from a deterministic variant of the reachability problem. Given a directed graph G and its two vertices a and b , this problem is to determine whether there is a deterministic path from a to b . A deterministic path is such that for every edge (i, j) in the path there is no other edge in G going out of i . Similarly to above, and with respect to first-order reductions, w.l.o.g. we may assume that the out-degrees of a and b are 1 and G has no cycles except for a self-loop on b . An instance of deterministic reachability is reduced to a team X over $\{x, y\}$ by applying the formation rule:

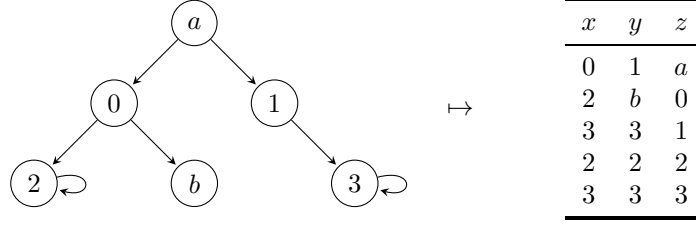


Figure 4: Reachability and $\text{MSM}(x \subseteq z \wedge y \subseteq z)$ over teams with a key z

- For each edge (u, v) that is the only edge going out from u , add the assignment $(y, x) \mapsto (u, v)$ to X .

For instance, the instance illustrated in Fig. 1 reduces to the team that consists of the three assignments $(y, x) \mapsto \{(a, 0), (1, 2), (b, b)\}$. We show that the input instance admits positive answer iff $m \in \nu(X, x \subseteq y)$, where m is the assignment that corresponds to the only edge going out from s .

By Lemma 14, $s \in \nu(X, x \subseteq y)$ iff s is connected to a cycle in the graph $G_X = (X, E)$, where $(s_0, s_1) \in E$ iff $s_0(y) = s_1(x)$. Since the only cycle in G is the self-loop on b , accordingly the only cycle in G_X is the self-loop on the mapping that corresponds to the self-loop on b . Furthermore, the team formation rule for X implies that any path in G_X corresponds to a deterministic path in G , and vice versa. Consequently, $s \in \nu(X, x \subseteq y)$ iff there is a deterministic path from a to b .

Lastly, we note that the reduction to X and s is clearly first-order, and that y is a key on X . This concludes the hardness proof.

Membership. Since y is a key on X , all paths in the graph G given in Lemma 14 are deterministic. The existence of a path from a given assignment of X to a cycle in G is then confirmed if the unique path outgoing this assignment is of length at least $|X|$. This process can be clearly executed in **L**. □

Next we consider maximal subteam membership for a conjunction of two inclusion atoms which both refer to a key on the input team. Again, we utilize reachability for showing **NL**-hardness. In contrast to our earlier reduction from the same problem, we now have both an extra restriction and an extra allowance: teams must be of the sort where inclusion atoms point to some key, however instead of one inclusion atom now two inclusion atoms are available.

Theorem 25. $\text{MSM}(x \subseteq z \wedge y \subseteq z)$ over teams for which z is a key is **NL**-complete.

Proof. Hardness. As above, it suffices to give a logarithmic-space many-one reduction from reachability. Recall that reachability is the problem to decide, given a directed graph G with two nodes a and b , whether there is a path from a to b . W.l.o.g. we restrict attention to instances in which all vertices have out-degree at most two. Since **NL** is closed under complement, we may reduce to the complement of $\text{MSM}(x \subseteq z \wedge y \subseteq z)$ over teams with a key z . W.l.o.g. we may assume that b has out-degree 0 and that all the other nodes have out-degree at least 1. The team X over $\{x, y, z\}$ is now constructed by the following formation rule (see also Fig. 4). For all vertices $i \neq b$:

- if i has single outgoing edge (i, j) , add the assignment $s_i : (x, y, z) \mapsto (j, j, i)$ to X ;
- if i has two outgoing edges (i, j) and (i, k) , add the assignment $s_i : (x, y, z) \mapsto (j, k, i)$ to X .

Observe that z is a key on X . By Theorem 16 it suffices to show that Player II has no winning strategy in $\text{safety}(X, s_a, x \subseteq z \wedge y \subseteq z)$ iff a is connected to b .

First note that since z is a key, Player II has a pre-determined strategy: either there is no assignment to choose or else Player II has to always pick the only possible assignment which keeps the game running. For instance, given an assignment-node position (s, n) where n is labeled by $x \subseteq z$, Player II can only counter by selecting the position (n, s') where s' is the only assignment of X' such that $s'(y) = s(x)$, provided that such an assignment exists. Hence, a is connected to b iff Player I wins some instance of $\text{safety}(X, s_a, x \subseteq z \wedge y \subseteq z)$. Furthermore, Player I wins iff the game reaches a position (s, n) where s maps either x or u to b . A sequence of positions that is winning for Player I now generates a path from a to b , formed by following the edges which correspond to Player I's selections between $x \subseteq z$ and $y \subseteq z$.

Conversely, a winning sequence for Player I can be found by moving to $x \subseteq z$ ($y \subseteq z$, resp.) whenever the values of (z, x) ((z, y) , resp.) form the next edge on the path.

Membership. Since NL is closed under complement, it suffices to describe a NL procedure for deciding the complement problem for $\text{MSM}(x \subseteq z \wedge y \subseteq z)$ over teams with a key z . By Theorem 16 and Lemma 17, $s \notin \nu(X, x \subseteq z \wedge y \subseteq z)$ iff Player II has no winning strategy in $\text{safety}_{3,l}(X, s, x \subseteq z \wedge y \subseteq z)$, where $l = |X|$. Recall that z is a key on the given team X , and hence the strategy of Player II is pre-determined. Hence, it suffices to guess the choices of Player I and accept iff the game terminates before $3 \cdot l$ assignment-element pairs have been played in the game. \square

Adapting the reasoning behind Corollary 22 we obtain a corollary for disjunction, too.

Corollary 26. $\text{MSM}(x \subseteq z \vee y \subseteq z)$ over teams for which z is a key, is **L**-complete.

4.3 Consistent Query Answering

The maximal subteam 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 $\sigma = \{R_1, \dots, R_n\}$ 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 $q = \psi(x_1, \dots, x_n)$ on a database D is any (a_1, \dots, a_n) such that D satisfies $\psi(a_1, \dots, a_n)$, and a *consistent answer* on an inconsistent database I is any value (a_1, \dots, a_n) such that each repair D of I satisfies $\psi(a_1, \dots, a_n)$. Let $*$ $\in \{\oplus, \text{subset}, \text{superset}\}$ and let Σ be a set of integrity constraints. The **-repair checking problem w.r.t. Σ (*-RC(Σ))* 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 (*-CQA(Σ, q))* is to determine, given an inconsistent database I , whether q is true in every *-repair of I . Inclusion dependencies are a special case of so-called LAV tgds that are first-order formulae of the form $\forall \bar{x}(\psi(\bar{x}) \rightarrow \exists \bar{y}\theta(\bar{x}, \bar{y}))$ where ψ is a single relational atom and θ is a conjunction of relational atoms. For LAV tgds these problems are solvable in polynomial time.

Theorem 27 ([24]). *Let $*$ $\in \{\oplus, \text{subset}, \text{superset}\}$, 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.*

Furthermore, it is known that 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 19 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 $\Sigma \in IC$, *-CQA(Σ, q) is in C for all Boolean conjunctive queries q and C -complete for some such q .

Corollary 28. *Let $*$ $\in \{\oplus, \text{subset}\}$. 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.*

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 ; for the lower bounds note that in our reductions $s \in \nu(X, \phi)$ if and only if $\nu(X, \phi) \neq \emptyset$.² Considering consistent query answering, for the upper bounds we may apply the maximal subteam membership problem and for lower bounds we may simply use atomic queries. That we may include also \oplus -repairs 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 [24]. 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.

5 Complexity of 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 29. Let $\phi \in \text{FO}(\subseteq)$. Then $\text{MC}(\phi)$ 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.

Lemma 30. Let $\alpha, \beta \in \text{FO}(\subseteq)$ be such that

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

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

Proof. Let (\mathfrak{A}, X, s) be an instance of $\text{MSM}(\alpha)$, that is, \mathfrak{A} is a model, X a team over $\text{Fr}(\alpha)$ and $s \in X$. It suffices to define a first-order reduction from (\mathfrak{A}, X, s) to a team X' over $\text{Fr}(\alpha) \cup \text{Fr}(\beta)$ 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 $\text{Fr}(\alpha) \cup \text{Fr}(\beta)$. 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 $\nu(\mathfrak{A}, X', \alpha) \upharpoonright \text{Fr}(\alpha) = \nu(\mathfrak{A}, X' \upharpoonright \text{Fr}(\alpha), \alpha) = \nu(\mathfrak{A}, X, \alpha)$, and similarly $\nu(\mathfrak{A}, X', \beta) \upharpoonright \text{Fr}(\beta) = \nu(\mathfrak{A}, Y, \beta) = Z_0$. 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 $s \in \nu(\mathfrak{A}, X, \alpha)$. \square

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 32(??), it is strictly less hard.

Lemma 31. Let $\alpha \in \text{FO}(\subseteq)$ be such that $\text{MSM}(\alpha)$ is in $C \in \{\mathbf{L}, \mathbf{NL}\}$. Then $\text{MC}(\alpha)$ is in C .

By Lemmata 13, 30, 31, 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 32.

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

²In point of fact, the reduction of Lemma 21 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.

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(\subseteq). By Lemma 13, we already know that NL-computability of maximal subteam membership is preserved under disjunctions. However, Theorem 19 shows that conjunction can increase the complexity of the maximal subteam membership problem from NL to P-complete, and by Proposition 32, 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 $\text{Gm}(V, E, a)$ 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 $\text{Gm}(V, E, a)$.

Note that GAME can be seen as a variation of the monotone circuit value problem MCVP. Indeed, it is straightforward to define for a given monotone circuit C and input word w an input (V, E, a) for GAME such that $\text{Gm}(V, E, a)$ 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 33. *Let ϕ be the formula $\forall z(\neg Eyz \vee z \subseteq x)$. Then $\text{MSM}(\phi)$ is P-complete. Consequently, $\text{MC}(\phi \vee Euv)$ is also P-complete.*

Proof. We give now a reduction from GAME to $\text{MSM}(\phi)$. 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 $\mathfrak{A} = (V, E)$, $X = \{s : \{x, y\} \rightarrow V \mid (s(x), s(y)) \in E\}$ 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 $\text{Gm}(V, E, c)$ 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 $\text{Gm}(V, E, c)$.

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 $Z_0 = \{s \in Z \mid s(z) = s(x) \text{ and } s(y) \in I\}$. 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 $\text{Gm}(V, E, s(y))$ 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 $\text{Gm}(V, E, s(y))$. 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.

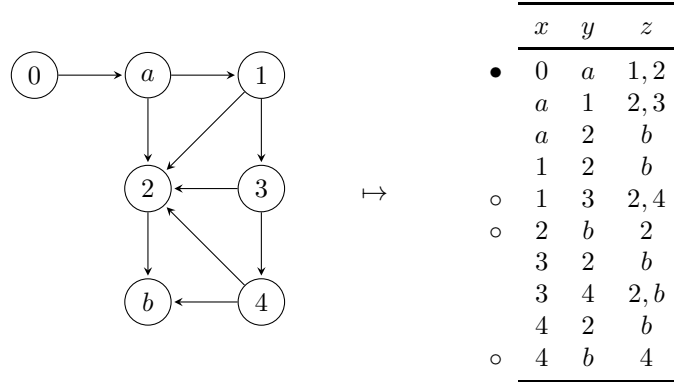


Figure 5: GAME and MSM($\forall z(\neg Eyz \vee z \subseteq x)$)

Using Lemma 30, we see that $\text{MC}(\forall z(\neg Eyz \vee z \subseteq x) \vee \beta)$ is **P**-hard for any non-trivial formula β such that $x, y \notin \text{Fr}(\beta)$, in particular for $\beta = Euv$. \square

The example above shows 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 34. *Let $\phi \in \text{FO}(\subseteq)$, $\psi \in \text{FO}$, 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 $\mathfrak{A} \models_Y \exists x\phi$. Then there is a function $F : X \rightarrow \mathcal{P}(A) \setminus \{\emptyset\}$ 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 $G : X'' \rightarrow \mathcal{P}(A) \setminus \{\emptyset\}$ 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. 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 $X'' = \nu(\mathfrak{A}, X', \phi)$. 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 $X'' \subseteq \nu(\mathfrak{A}, X, \phi \wedge \psi)$. \square

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

Proposition 35. *Let $\phi \in \text{FO}(\subseteq)$, $\psi \in \text{FO}$, and assume that $\text{MSM}(\phi)$ is in a complexity class $C \in \{\mathbf{L}, \mathbf{NL}\}$. Then*

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

Proof. (a) By Lemma 34(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 $\text{MSM}(\phi)$ is in C .

(b) By Lemma 34(b), 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 $\psi \in \text{FO}$, the team X' can be computed in C , whence the claim follows from the assumption that $\text{MSM}(\phi)$ is in C . \square

Summarising Lemma 13 and Proposition 35, **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 20, for all inclusion atoms, we define our fragment $\text{FO}(\subseteq)_w$ of inclusion logic by the following grammar:

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

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

Theorem 36. $\text{MC}(\phi)$ is in **NL** for every $\phi \in \text{FO}(\subseteq)_w$.

Proof. By an easy induction we see that $\text{MSM}(\phi)$ is in **NL** for every $\phi \in \text{FO}(\subseteq)_w$. The claim follows now from Lemma 31. \square

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

Theorem 37. Over finite ordered models, $\text{TC} \leq \text{FO}(\subseteq)_w$.

Proof. By Theorem 9 we may assume without loss of generality that any **TC** sentence ϕ is of the form $[\text{TC}_{\bar{x}, \bar{y}} \alpha(\bar{x}, \bar{y})](\overline{\min}, \overline{\max})$ where \bar{x} and \bar{y} are n -ary sequences of variables. We define an equivalent $\text{FO}(\subseteq)_w$ sentence ϕ' as follows:

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

where

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

For two tuples of variables \bar{x} and \bar{y} of the same length, we write $\bar{x} < \bar{y}$ as a shorthand for the formula expressing that \bar{x} is less than \bar{y} in the induced lexicographic ordering, and $\bar{x} = \bar{y}$ for the conjunction expressing that \bar{x} and \bar{y} are pointwise identical. 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 \bar{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 \bar{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 $s'(\bar{x})$ for some $s' \in X$ with $s'(\bar{t}_x) = \overline{\max}$. Lastly, by $\mathfrak{A} \models_X \psi_4$ it follows that $s'(\bar{x}) = \overline{\max}$ which shows that $[\text{TC}_{\bar{x}, \bar{y}} \alpha(\bar{x}, \bar{y})](\overline{\min}, \overline{\max})$.

Assume then that $[\text{TC}_{\bar{x}, \bar{y}} \alpha(\bar{x}, \bar{y})](\overline{\min}, \overline{\max})$, that is, there is an α -path $\bar{v}_1, \dots, \bar{v}_k$ where $\bar{v}_1 = \overline{\min}$ and $\bar{v}_k = \overline{\max}$. We may assume that there are no cycles in the path. Let \bar{a}_i denote the i th element in the lexicographic ordering of A^n . We let $X = \{s_1, \dots, s_k\}$ be such that $(\bar{x}, \bar{y}, \bar{t}_x, \bar{t}_y)$ is mapped to $(\bar{v}_i, \bar{v}_{i+1}, \bar{a}_i, \bar{a}_{i+1})$ by s_i , for $i = 1, \dots, k-2$, to $(\bar{v}_{k-1}, \bar{v}_k, \bar{a}_{k-1}, \overline{\max})$ by s_{k-1} , and to $(\bar{v}_k, \bar{v}_1, \overline{\max}, \overline{\min})$ 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 $\mathfrak{A} \models \phi'$. \square

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

Theorem 38. A class \mathcal{C} of finite ordered models is in **NL** iff it can be defined in $\text{FO}(\subseteq)_w$.

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 also presented a safety game for the maximal subteam problem and used it to demonstrate that the problem is less complex if the range of inputs is restricted to teams on which the inclusion atoms reference a key. 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.

References

- [1] Foto N. Afrati and Phokion G. Kolaitis. Repair checking in inconsistent databases: algorithms and complexity. In *Database Theory - ICDT 2009, 12th International Conference, St. Petersburg, Russia, March 23-25, 2009, Proceedings*, pages 31–41, 2009. URL: <http://doi.acm.org/10.1145/1514894.1514899>, doi:10.1145/1514894.1514899.
- [2] Marcelo Arenas, Leopoldo E. Bertossi, and Jan Chomicki. Consistent query answers in inconsistent databases. In *Proceedings of the Eighteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, May 31 - June 2, 1999, Philadelphia, Pennsylvania, USA*, pages 68–79, 1999. URL: <http://doi.acm.org/10.1145/303976.303983>, doi:10.1145/303976.303983.
- [3] Jan Chomicki and Jerzy Marcinkowski. Minimal-change integrity maintenance using tuple deletions. *Inf. Comput.*, 197(1-2):90–121, 2005. URL: <https://doi.org/10.1016/j.ic.2004.04.007>, doi:10.1016/j.ic.2004.04.007.
- [4] Jukka Corander, Antti Hyttinen, Juha Kontinen, Johan Pensar, and Jouko Väänänen. A logical approach to context-specific independence. In *Logic, Language, Information, and Computation - 23rd International Workshop, WoLLIC 2016, Puebla, Mexico, August 16-19th, 2016. Proceedings*, pages 165–182, 2016. URL: https://doi.org/10.1007/978-3-662-52921-8_11, doi:10.1007/978-3-662-52921-8_11.
- [5] Arnaud Durand, Miika Hannula, Juha Kontinen, Arne Meier, and Jonni Virtema. Approximation and dependence via multiteam semantics. *Annals of Mathematics and Artificial Intelligence*, Jan 2018. URL: <https://doi.org/10.1007/s10472-017-9568-4>, doi:10.1007/s10472-017-9568-4.
- [6] Arnaud Durand, Juha Kontinen, Nicolas de Rugy-Altherre, and Jouko Väänänen. 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.*, pages 73–85, 2015. URL: <https://doi.org/10.4204/EPTCS.193.6>, doi:10.4204/EPTCS.193.6.
- [7] Johannes Ebbing, Lauri Hella, Arne Meier, Julian-Steffen Müller, Jonni Virtema, and Heribert Vollmer. Extended modal dependence logic. In *Logic, Language, Information, and Computation - 20th International Workshop, WoLLIC 2013, Darmstadt, Germany, August 20-23, 2013. Proceedings*, pages 126–137, 2013. URL: http://dx.doi.org/10.1007/978-3-642-39992-3_13, doi:10.1007/978-3-642-39992-3_13.
- [8] Johannes Ebbing, Juha Kontinen, Julian-Steffen Müller, and Heribert Vollmer. A fragment of dependence logic capturing polynomial time. *Logical Methods in Computer Science*, 10(3), 2014. URL: [http://dx.doi.org/10.2168/LMCS-10\(3:3\)2014](http://dx.doi.org/10.2168/LMCS-10(3:3)2014), doi:10.2168/LMCS-10(3:3)2014.
- [9] Pietro Galliani. Inclusion and exclusion dependencies in team semantics: On some logics of imperfect information. *Annals of Pure and Applied Logic*, 163(1):68 – 84, 2012.
- [10] Pietro Galliani and Lauri Hella. Inclusion Logic and Fixed Point Logic. In Simona Ronchi Della Rocca, editor, *Computer Science Logic 2013 (CSL 2013)*, volume 23 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 281–

- 295, Dagstuhl, Germany, 2013. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. URL: <http://drops.dagstuhl.de/opus/volltexte/2013/4203>, doi:<http://dx.doi.org/10.4230/LIPIcs.CSL.2013.281>.
- [11] Erich Grädel. Model-checking games for logics of imperfect information. *Theoretical Computer Science (to appear)*, 2012. URL: <http://www.sciencedirect.com/science/article/pii/S0304397512009541>, doi:[10.1016/j.tcs.2012.10.033](https://doi.org/10.1016/j.tcs.2012.10.033).
- [12] Erich Grädel. Games for inclusion logic and fixed-point logic. In *Dependence Logic, Theory and Applications*, pages 73–98. 2016. URL: https://doi.org/10.1007/978-3-319-31803-5_5, doi:[10.1007/978-3-319-31803-5_5](https://doi.org/10.1007/978-3-319-31803-5_5).
- [13] Miika Hannula and Juha Kontinen. A finite axiomatization of conditional independence and inclusion dependencies. *Inf. Comput.*, 249:121–137, 2016. URL: <https://doi.org/10.1016/j.ic.2016.04.001>, doi:[10.1016/j.ic.2016.04.001](https://doi.org/10.1016/j.ic.2016.04.001).
- [14] Miika Hannula, Juha Kontinen, and Jonni Virtema. Polyteam semantics. In *Logical Foundations of Computer Science - International Symposium, LFCS 2018, Deerfield Beach, FL, USA, January 8-11, 2018, Proceedings*, pages 190–210, 2018. URL: https://doi.org/10.1007/978-3-319-72056-2_12, doi:[10.1007/978-3-319-72056-2_12](https://doi.org/10.1007/978-3-319-72056-2_12).
- [15] Miika Hannula, Juha Kontinen, Jonni Virtema, and Heribert Vollmer. Complexity of propositional logics in team semantic. *ACM Trans. Comput. Log.*, 19(1):2:1–2:14, 2018. URL: <http://doi.acm.org/10.1145/3157054>, doi:[10.1145/3157054](https://doi.org/10.1145/3157054).
- [16] Wilfrid Hodges. Compositional Semantics for a Language of Imperfect Information. *Journal of the Interest Group in Pure and Applied Logics*, 5 (4):539–563, 1997.
- [17] Neil Immerman. Relational queries computable in polynomial time. *Information and control*, 68(1):86–104, 1986.
- [18] Neil Immerman. Languages that capture complexity classes. *SIAM J. Comput.*, 16(4):760–778, 1987.
- [19] Neil Immerman. Nondeterministic space is closed under complementation. *SIAM J. Comput.*, 17(5):935–938, 1988. URL: <https://doi.org/10.1137/0217058>, doi:[10.1137/0217058](https://doi.org/10.1137/0217058).
- [20] Jarmo Kontinen. Coherence and computational complexity of quantifier-free dependence logic formulas. *Studia Logica*, 101(2):267–291, 2013. URL: <http://dx.doi.org/10.1007/s11225-013-9481-8>, doi:[10.1007/s11225-013-9481-8](https://doi.org/10.1007/s11225-013-9481-8).
- [21] Juha Kontinen, Antti Kuusisto, Peter Lohmann, and Jonni Virtema. Complexity of two-variable dependence logic and if-logic. *Inf. Comput.*, 239:237–253, 2014. URL: <https://doi.org/10.1016/j.ic.2014.08.004>, doi:[10.1016/j.ic.2014.08.004](https://doi.org/10.1016/j.ic.2014.08.004).
- [22] Paraschos Koutris and Jef Wijsen. Consistent query answering for self-join-free conjunctive queries under primary key constraints. *ACM Trans. Database Syst.*, 42(2):9:1–9:45, 2017. URL: <http://doi.acm.org/10.1145/3068334>, doi:[10.1145/3068334](https://doi.org/10.1145/3068334).
- [23] Eric Pacuit and Fan Yang. *Dependence and Independence in Social Choice: Arrow’s Theorem*, pages 235–260. Springer International Publishing, Cham, 2016. URL: https://doi.org/10.1007/978-3-319-31803-5_11, doi:[10.1007/978-3-319-31803-5_11](https://doi.org/10.1007/978-3-319-31803-5_11).

- [24] Balder ten Cate, Gaëlle Fontaine, and Phokion G. Kolaitis. On the data complexity of consistent query answering. *Theory Comput. Syst.*, 57(4):843–891, 2015. URL: <https://doi.org/10.1007/s00224-014-9586-0>, doi:10.1007/s00224-014-9586-0.
- [25] Jouko Väänänen. *Dependence Logic*. Cambridge University Press, 2007.
- [26] Jouko Väänänen. Modal Dependence Logic. In Krzysztof R. Apt and Robert van Rooij, editors, *New Perspectives on Games and Interaction*. Amsterdam University Press, Amsterdam, 2008.
- [27] Moshe Y Vardi. The complexity of relational query languages. In *Proceedings of the fourteenth annual ACM symposium on Theory of computing*, pages 137–146. ACM, 1982.
- [28] Jonni Virtema. Complexity of validity for propositional dependence logics. *Inf. Comput.*, 253:224–236, 2017. URL: <https://doi.org/10.1016/j.ic.2016.07.008>, doi:10.1016/j.ic.2016.07.008.
- [29] Heribert Vollmer. *Introduction to Circuit Complexity - A Uniform Approach*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 1999. URL: <https://doi.org/10.1007/978-3-662-03927-4>, doi:10.1007/978-3-662-03927-4.
- [30] Fan Yang and Jouko Väänänen. Propositional logics of dependence. *Ann. Pure Appl. Logic*, 167(7):557–589, 2016. URL: <https://doi.org/10.1016/j.apal.2016.03.003>, doi:10.1016/j.apal.2016.03.003.