Information and Computation ••• (••••) ••••••

Contents lists available at ScienceDirect

Information and Computation

www.elsevier.com/locate/yinco

Complexity thresholds in inclusion logic

Miika Hannula [a,*], Lauri Hella [b]

[a] University of Helsinki, Department of Mathematics and Statistics, P.O. Box 64, 00014 Helsinki, Finland
[b] Tampere University, Faculty of Information Technology and Communication Sciences, P.O. Box 607, 33014 Tampere, Finland

ARTICLE INFO

Article history:
Available online xxxx

Keywords:
Team semantics
Inclusion logic
Complexity
Consistent query answering

ABSTRACT

Inclusion logic differs from many other logics of dependence and independence in that it can only describe polynomial-time properties. In this article we examine more closely connections between syntactic fragments of inclusion logic and different complexity classes. Our focus is on 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. We also present a safety game for the maximal subteam membership problem and use it to investigate this problem over teams in which one variable is a key. Furthermore, we relate our findings to consistent query answering over inclusion dependencies, and present a fragment of inclusion logic that captures non-deterministic logarithmic space in ordered models.

© 2021 The Author(s). Published by Elsevier Inc. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

1. Introduction

In this article we study the computational complexity of inclusion logic. Inclusion logic was introduced by Galliani [11] as a variant of dependence logic, developed by Väänänen in 2007 [30]. Dependence logic adds to first-order logic novel dependence atoms $\mathrm{dep}(\overline{x}, y)$ expressing that a variable y depends only on variables in $\overline{x}$. Inclusion logic, instead, adds inclusion atoms $\overline{x} \subseteq \overline{y}$ indicating that all values of $\overline{x}$ appear also as values of $\overline{y}$. 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 [20,33,35], modal logic [8,21,31], probabilistic logics [6], and two-variable logics [27]. Recent research has also pursued connections and applications of dependence logic to fields such as database theory [18,19], Bayesian networks [5], and social choice theory [29]. A common notion underlying all these endeavours is that of team semantics. Team semantics, introduced by Hodges in [22], is a semantic 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 sets of assignments, probability distributions, or database tables, each having their characteristic versions of team semantics [30,6,19].

After the introduction of dependence logic Grädel and Väänänen observed that team semantics is also suitable for a logic of independence [14]. Then, Galliani introduced inclusion logic and other logical formalisms inspired by dependency concepts in database theory [11]. Shortly after, Galliani and Hella showed that inclusion logic is equi-expressive to positive

* Corresponding author.
E-mail addresses: miika.hannula@helsinki.fi (M. Hannula), lauri.hella@tuni.fi (L. Hella).

https://doi.org/10.1016/j.ic.2021.104759
0890-5401/© 2021 The Author(s). Published by Elsevier Inc. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

Please cite this article as: M. Hannula and L. Hella, Complexity thresholds in inclusion logic, Information and Computation, https://doi.org/10.1016/j.ic.2021.104759

greatest-fixed point logic [13], thus establishing inclusion logic as the first team-based logic to capture polynomial time (**P**) (over ordered structures). Starting from this general result, our aim here is at developing a more thorough understanding of the complexity of inclusion logic. In particular, the current paper identifies several complexity thresholds with respect to two key computational problems: maximal subteam membership and model checking.

Inclusion logic differs from other team-based logics in that it is closed under unions of teams. This implies that for any team X and any inclusion logic formula ϕ, there is a unique maximal subteam of X satisfying ϕ. The maximal subteam membership problem $\mathsf{MSM}(\phi)$, for a fixed formula ϕ, now asks whether a given assignment appears in this maximal subteam for a given team X. 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 [4] 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 [3].

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. [1,4,28,3]. 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 of inclusion logic 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 data complexity of model checking in team semantics has been studied in [7,26] for dependence and independence logics. For these logics increase in complexity arises particularly from disjunctions. For example, the data complexity of model checking for a disjunction of three or two dependence atoms is respectively complete for **NP** and non-deterministic logarithmic space (**NL**), while a single dependence atom is first-order definable [26]. 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 **NL**-complete 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 **P**-complete. This result gives a complete characterization of the maximal subteam membership problem for conjunctions of unary inclusion atoms. Based on it we also prove results on data complexity 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 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 logarithmic space (**L**) and **NL** under these assumptions.

Finally, restricting to finite ordered structures, we present a fragment of inclusion logic that corresponds to **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** [9]. 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 [24,25].

The present article is the full version of the conference article [17].

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 $\overline{q}$.

We assume that the reader is familiar with standard concepts in 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. Throughout the paper, all {**NL**, **P**}-completeness results are stated under logarithmic space many-one reductions, and all **L**-completeness results under first-order many-one reductions. Further, in this paper we consider model checking problems only with respect to fixed formulae. The complexity of such a problem $\mathsf{MC}(\phi)$ is also commonly referred to as data complexity.

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, which may also contain function symbols. 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 over $\mathfrak{A}$ 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 over $\mathfrak{A}$ 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 $\{s \upharpoonright V \mid s \in X\}$. Let X be a team. For a mapping $F : X \to \mathcal{P}(A) \setminus \{\emptyset\}$, we define

$$X[F/x] := \{s(a/x) \mid s \in X, a \in F(s)\}.$$

For a set B, we define

$$X[B/x] := \{s(b/x) \mid s \in X, b \in B\}.$$

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 for all $s \in X : \mathfrak{A} \models_s \alpha$, 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 $\emptyset$.

In this paper, we focus on computational problems related to formulae of inclusion logic. On the other hand, we also investigate the expressivity of inclusion logic at the level of *sentences*. 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. We also write $\mathcal{L}_1 \equiv \mathcal{L}_2$ if $\mathcal{L}_1 \leq \mathcal{L}_2$ and $\mathcal{L}_1 \geq \mathcal{L}_2$; and $\mathcal{L}_1 < \mathcal{L}_2$ if $\mathcal{L}_1 \leq \mathcal{L}_2$ but $\mathcal{L}_1 \not\geq \mathcal{L}_2$.

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

Proposition 2 *(Flatness [30]). 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 $\bar{x}$ 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:

$$\mathfrak{A} \models_X \bar{x} \subseteq \bar{y} \text{ if for all } s \in X \text{ there is } s' \in X \text{ such that } s(\bar{x}) = s'(\bar{y}).$$

In other words, an inclusion atom $\overline{x} \subseteq \overline{y}$ holds if the values of $\overline{y}$ contain the values of $\overline{x}$ in X, that is,

$$\mathfrak{A} \models_X \overline{x} \subseteq \overline{y} \iff X(\overline{x}) \subseteq X(\overline{y}),$$

where $X(\overline{z}) := \{s(\overline{z}) \mid s \in X\}$, for a tuple of variables $\overline{z}$. 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. These properties of inclusion logic can be sensitive to the choice of semantics used. With respect to so-called *strict team semantics* [11] inclusion logic is not anymore local nor union closed; in fact, it then corresponds to **NP** over sentences [12].

Proposition 4 *(Locality [11]). 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*

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

Proposition 5 *(Union closure [11]). 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 $\bigcup\{X \mid \mathfrak{A} \models_X \phi\}$ is the unique maximal team over $\mathfrak{A}$ satisfying ϕ. Further, union closure implies the *empty team property*, that is, $\mathfrak{A} \models_\emptyset \phi$ for all inclusion logic formulae ϕ. Similarly, downward closure implies the empty team property. Yet these closure properties are not necessary conditions for the empty team property; for instance, independence logic has the empty team property but is neither union, nor downward closed.

The starting point for our investigations is the result by Galliani and Hella [13] characterizing the expressivity of inclusion logic in terms of positive greatest fixed point logic. The latter logic is obtained from greatest 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 [23] and Vardi [32] that inclusion logic captures polynomial time in finite ordered models.

Theorem 6 *([13]). Every inclusion logic sentence is equivalent to some positive greatest fixed point logic sentence, and vice versa.*

Theorem 7 *([13]). A class $\mathcal{C}$ of finite ordered models is in* **P** *iff it can be defined in* $\mathrm{FO}(\subseteq)$.

2.3. Transitive closure logic

In Section 6 we relate inclusion logic to transitive closure logic, and hence we next give a short introduction to the latter. A binary relation R is said to be *transitive* if $(a, b) \in R$ and $(b, c) \in R$ imply $(a, c) \in R$. The *transitive closure* of a binary relation R, written $\mathrm{TC}(R)$, is defined as the intersection of all binary 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 $(a, b) \in R_i$ if and only if there is an R-path of length $i + 1$ (i.e., containing $i + 1$ edges) from a to 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 binary relation defined as follows:

$$R_{\psi,\mathfrak{A},s} = \{(\overline{a}, \overline{b}) \in (M^k)^2 \mid \mathfrak{A} \models_{s(\overline{a}/\overline{x}, \overline{b}/\overline{y})} \psi\}.$$

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. Furthermore, for a sequence of terms $\overline{t} = (t_1, \ldots, t_n)$, we write $\overline{t}^{\mathfrak{A},s} = (t_1^{\mathfrak{A},s}, \ldots, t_n^{\mathfrak{A},s})$.

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 *([24,25]). 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. We also define a safety game for quantifier-free inclusion logic formulae. This game will be used to facilitate complexity analysis in this section. The section is organized as follows. In Sections 3.1 and 3.2 we present some basic properties of the maximal subteam membership problem and the safety game, respectively. In Section 3.3 the complexity of the maximal subteam membership problem is studied, and in Section 3.4 we restrict attention to cases where inclusion atoms are thought of as foreign keys.

3.1. Definition and basic properties

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 $\mathfrak{A}\models_Y\phi$. Suppose ϕ does not contain quantifiers or symbols from the underlying vocabulary, that is, the truth value of ϕ only depends on the team. 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 $s\in X$.

We obtain a polynomial-time upper bound for maximal subteam membership from the following result by Grädel (see Theorem 24 in [15]). We write $\mathrm{rel}(X):=\{(a_1,\ldots,a_n)\in\mathrm{Dom}(\mathfrak{A})^n\mid s(x_1,\ldots,x_n)=(a_1,\ldots,a_n),s\in X\}$ for the relational encoding of a team X with domain $\{x_1,\ldots,x_n\}$.

Theorem 11 *([15]). For any* FO($\subseteq$)*-formula $\psi(\overline{x})$, there is a formula $\phi(R,\overline{x})$ of positive greatest fixed point logic, with only positive occurrences of R, such that for any structure $\mathfrak{A}$ and team X,*

$$\mathfrak{A}\models_X\psi(\overline{x})\iff(\mathfrak{A},\mathrm{rel}(X))\models\forall\overline{x}(R\overline{x}\to\phi(R,\overline{x}))$$

In particular, for all assignments s, we have that $\mathfrak{A}\models_s[\mathrm{GFP}_{R,\overline{x}}\phi(R,\overline{x})](\overline{x})$ if and only if $s\in\bigcup\{X\mid\mathfrak{A}\models_X\psi(\overline{x})\}$.

A straightforward adaptation of Theorem 11 shows that $\nu(\mathfrak{A},X,\phi)$ is also definable in positive greatest fixed point logic. Namely, letting $\phi'(R,S,\overline{x}):=\phi(R,\overline{x})\wedge S(\overline{x})$ we obtain that

$$\mathfrak{A}\models_X\psi(\overline{x})\text{ and }X\subseteq Y\iff(\mathfrak{A},\mathrm{rel}(X),\mathrm{rel}(Y))\models\forall\overline{x}(R\overline{x}\to\phi'(R,S,\overline{x})).$$

Consequently, for all assignments s and $\theta(S,\overline{x}):=[\mathrm{GFP}_{R,\overline{x}}\phi(R,S,\overline{x})](\overline{x})$, we have that

$$(\mathfrak{A},\mathrm{rel}(Y))\models_s\theta(S,\overline{x})\iff s\in\bigcup\{X\subseteq Y\mid\mathfrak{A}\models_X\psi(\overline{x})\}=\nu(\mathfrak{A},Y,\psi).$$

For an introduction of the greatest fixed point logic we refer the reader to [10].

Thus, it follows that maximal subteam membership problem is polynomial time computable.

Lemma 12. *For every formula $\phi\in\mathrm{FO}(\subseteq)$, $\mathsf{MSM}(\phi)$ is in* **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 13. *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. Consider first "$\subseteq$". By definition there are subteams $Y, Z \subseteq X$ such that $Y \cup Z = v(\mathfrak{A}, X, \alpha \vee \beta)$, $Y \models \alpha$ and $Z \models \beta$. For all formulae ϕ, $v(\mathfrak{A}, X, \phi)$ is the union of all subteams $X' \subseteq X$ such that $\mathfrak{A} \models_{X'} \phi$. Thus $Y \subseteq v(\mathfrak{A}, X, \alpha)$ and $Z \subseteq v(\mathfrak{A}, X, \beta)$, and consequently, $v(\mathfrak{A}, X, \alpha \vee \beta) \subseteq v(\mathfrak{A}, X, \alpha) \cup v(\mathfrak{A}, X, \beta)$.

Consider then "$\supseteq$". By definition $v(\mathfrak{A}, X, \alpha) \cup v(\mathfrak{A}, X, \beta)$ is a subteam of X satisfying $\alpha \vee \beta$. Consequently, $v(\mathfrak{A}, X, \alpha) \cup v(\mathfrak{A}, X, \beta) \subseteq v(\mathfrak{A}, X, \alpha \vee \beta)$. □

The previous lemma demonstrates that disjunction over two inclusion logic formulae does not increase the complexity of the maximal subteam problem. Indeed, to decide whether an assignment s is in $\mathsf{MSM}(\alpha \vee \beta)$, it suffices to decide (successively) whether s is in $\mathsf{MSM}(\alpha)$ or $\mathsf{MSM}(\beta)$. Thus we immediately obtain the following lemma.

Lemma 14. *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 s then belongs to the maximal subteam for $\overline{x} \subseteq \overline{y}$ if and only if there is a path from s to a cycle.[1] We say that a graph contains a path from a node v_0 to a cycle if it contains a path $(v_0, \ldots, v_n)$, where $(v_l, \ldots, v_n)$ is a cycle, for some $l < n$. Similarly, it contains a graph from a cycle to a node v_n if it contains a path $(v_0, \ldots, v_n)$, where $(v_0 \ldots, v_l)$ is a cycle, for some $0 < l$.

Lemma 15. *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 in 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) $s \in v(\mathfrak{A}, X, \overline{x} \subseteq \overline{y} \wedge \alpha)$ $\iff$ *G contains a path from s to a cycle,*

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

Proof. (a) Assume that $s \in v(\mathfrak{A}, X, \overline{x} \subseteq \overline{y} \wedge \alpha)$. 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 $s''(\overline{y}) = s'(\overline{x})$. Also, since $\mathfrak{A} \models_Y \alpha$, it follows by downward closure of α that $\mathfrak{A} \models_{\{s',s''\}} \alpha$, whence $(s', s'') \in E$. In particular there is a non-ending path in G starting from s. Since X is finite, there must be a path from s to a cycle. Conversely, assume G contains a path from s to a cycle. Then $\mathfrak{A} \models_Y \overline{x} \subseteq \overline{y}$, where Y consists of all assignments in the path and the cycle. Furthermore, since α is union closed and $\mathfrak{A} \models_{\{s,s'\}} \alpha$ for all edges in the path and the cycle, it follows that $\mathfrak{A} \models_Y \alpha$. Hence, $s \in v(\mathfrak{A}, X, \overline{x} \subseteq \overline{y} \wedge \alpha)$.

(b) It follows from (a) that $s \in v(\mathfrak{A}, X, \overline{y} \subseteq \overline{x} \wedge \alpha)$ if and only if $G' = (X, E^{-1})$ contains a path from s to a cycle. Observe that a path in G' from s to a cycle constitutes, when reversed, a path from a cycle to s in G. Observe also that $v(\mathfrak{A}, X, \phi \wedge \psi) \subseteq v(\mathfrak{A}, X, \phi)$ for all inclusion logic formulae ϕ and ψ, because each subteam of X satisfying $\phi \wedge \psi$ also satisfies ϕ. Hence, if $s \in v(\mathfrak{A}, X, \overline{x} \subseteq \overline{y} \wedge \overline{y} \subseteq \overline{x} \wedge \alpha)$, then G contains a path from a cycle to s and another path from s to a cycle, i.e., a path from a cycle to another (or the same) cycle via s. Conversely, if such a path exists, then $\mathfrak{A} \models_Y \overline{x} \subseteq \overline{y} \wedge \overline{y} \subseteq \overline{x} \wedge \alpha$, where Y consists of all assignments in the path and the cycle(s). Consequently, $s \in v(\mathfrak{A}, X, \overline{x} \subseteq \overline{y} \wedge \overline{y} \subseteq \overline{x} \wedge \alpha)$. □

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 [15]. We present a safety game for a quadruple $(\mathfrak{A}, X, s, \phi)$, written $\mathsf{safety}(\mathfrak{A}, X, s, \phi)$, where s is an assignment in a team X, and ϕ is a quantifier-free formula. The main result of the section shows that the maximal subteam problem $\mathsf{MSM}(\phi)$ 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 $\phi \in \mathrm{FO}(\subseteq)$ with its *syntax tree*, that is, 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 $\mathsf{safety}(\mathfrak{A}, X, s, \phi)$ has two players: Player I intuitively argues that s is not in $v(\mathfrak{A}, X, \phi)$, while Player II tries to show the opposite. Each position of the game is associated with a node n of T_ϕ and an assignment s. If n is not a leaf, the game proceeds to a position associated with its child. If n is a leaf labeled by a first-order atom α, Player II wins if s satisfies α, and otherwise Player I wins. If n is labeled by an inclusion atom β, Player I either replaces n with its ancestor n', or forces Player II to replace s with another assignment s'. The prerequisite is that s' must validate β for s; if no such assignment exists, Player I wins. Further, if Player II can keep the game perpetually running, then she also wins.

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

Definition 16 *(Safety game).* Let $\phi \in \mathrm{FO}(\subseteq)$ be quantifier-free, and let s_0 be an assignment in a team X over a model $\mathfrak{A}$. The *safety game* $\mathsf{safety}(\mathfrak{A}, X, s_0, \phi)$ 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 first-order 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 $\overline{x} \subseteq \overline{y}$, then the game ends if there is no $s' \in X$ such that $s(\overline{x}) = s'(\overline{y})$. 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 $s(\overline{x}) = s'(\overline{y})$.

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 $\overline{x} \subseteq \overline{y}$.
- $\pi((n, s)) \in \{(s', n) \mid s' \in X, s(\overline{x}) = s'(\overline{y})\}$, for a leaf n labeled by $\overline{x} \subseteq \overline{y}$.

Player $A \in \{\mathrm{I}, \mathrm{II}\}$ has a winning strategy for $\mathsf{safety}(\mathfrak{A}, X, s_0, \phi)$ 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, except for nodes labeled by inclusion atoms, each node n is associated with either Player I or II. Since both players are active at positions associated with inclusion atoms, we use position ordering to denote which player is currently active. Also note that if ϕ does not contain any symbols from the underlying vocabulary, the outcome of $\mathsf{safety}(\mathfrak{A}, X, s, \phi)$ is independent of $\mathfrak{A}$, and thus we write $\mathsf{safety}(X, s, \phi)$ instead.

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

Theorem 17. *Let $\phi \in \mathrm{FO}(\subseteq)$ be quantifier-free, and let s be an assignment in a team X over a model $\mathfrak{A}$. Then $s \in \nu(\mathfrak{A}, X, \phi)$ iff Player II has a winning strategy in* $\mathsf{safety}(\mathfrak{A}, X, s, \phi)$.

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 $\overline{x} \subseteq \overline{y}$, then (n, s) is mapped to any (s', n) such that $s(\overline{x}) = s'(\overline{y})$ and $s' \in X_n$.

It can be shown by induction that $s \in X_n$ at any position (s, n) or (n, s) of a game where Player II follows her strategy. It also then follows that the strategy itself is well defined. First, the claim holds at the base case, where $n = r$. For the induction step, consider a position (s, n), where n is labeled by an inclusion atom. Then, if Player I selects n' such that n is a descendant of n', we have that $s \in X_{n'}$ is implied by $s \in X_n$ and $X_n \subseteq X_{n'}$. The other cases for the induction step are straightforward. Since $s \in X_n$ for any position (s, n), the game does not end if n is labeled by $\overline{x} \subseteq \overline{y}$; if n is labeled by a first-order literal ψ, then by flatness we obtain from $\mathfrak{A} \models_{X_n} \psi$ that $\mathfrak{A} \models_s \psi$. Thus we conclude that Player II has a winning strategy.

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. First, if n is labeled by $\overline{x} \subseteq \overline{y}$ and $s \in X_n$, 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 $s(\overline{x}) = s'(\overline{y})$. Second, if n is labeled by a first-order literal, then this literal is satisfied by any $s \in X_n$. Therefore, $\mathfrak{A} \models_{X_n} \alpha$ for any node n labeled by an atomic formula α. 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 $s \in \nu(\mathfrak{A}, X, \phi)$. □

Given that X is finite, it makes sense to consider bounded length restrictions of the safety game. We let $\mathsf{safety}_k(\mathfrak{A}, X, s, \phi)$ denote the version of $\mathsf{safety}(\mathfrak{A}, X, s, \phi)$ in which the winning conditions are as in $\mathsf{safety}(\mathfrak{A}, X, s, \phi)$, except that once pairs of the form (s, n) (i.e., pairs whose left element is an assignment and right element a node) have been played k times, then Player II immediately wins. The next lemma will be useful later.

Lemma 18. *Let $\phi \in \mathrm{FO}(\subseteq)$ be quantifier-free and such that T_ϕ has k nodes, and let s be an assignment in a team X that is of size l. Then Player II has a winning strategy for $\mathsf{safety}(\mathfrak{A}, X, s, \phi)$ iff she has a winning strategy for $\mathsf{safety}_{k\cdot l}(\mathfrak{A}, X, s, \phi)$.*

Proof. Suppose Player II has a winning strategy for $\mathsf{safety}_{k\cdot l}(\mathfrak{A}, X, s, \phi)$. Then, following her strategy, Player II either wins $\mathsf{safety}(\mathfrak{A}, X, s, \phi)$ before $k \cdot l$ assignment-node moves have been played, or the game reaches $k \cdot l$ assignment-node moves. In the latter case, positions of the form (s, n), the starting position included, have occurred $k \cdot l + 1$ many times, and thus some position (s, n) has occurred twice. Every time such a repetition is encountered, we may assume that $\mathsf{safety}_{k\cdot l}(\mathfrak{A}, X, s, \phi)$ is continued from the first occurrence of (s, n). Since the strategy of Player II is safe within $\mathsf{safety}_{k\cdot l}(\mathfrak{A}, X, s, \phi)$, we conclude that either the game terminates in a winning position for Player II, or the game never terminates in which case Player II also wins. □

3.3. 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 19. 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 20. *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 21. $\mathsf{MSM}(x \subseteq y)$ *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. Indeed, 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.

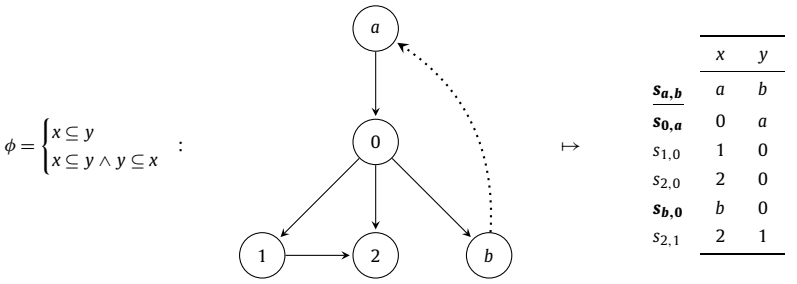
Define E' as the extension of E with an extra edge (b, a). Then b is reachable from a in G if and only if a belongs to a cycle in $G' = (V, E')$. We reduce from (G, a, b) to a team $X = \{s_{c,d} \mid (d, c) \in E'\}$ where $s_{u,v}$ maps (x, y) to (u, v) (see Fig. 1). By Lemma 15, b is reachable from 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 15 $\mathsf{MSM}(x \subseteq y)$ and $\mathsf{MSM}(x \subseteq y \wedge y \subseteq x)$ reduce to reachability variants that are clearly in **NL**. □

Next we turn to the third statement of Theorem 20. Recall that membership in **P** follows directly from Lemma 12. For **P**-hardness we reduce from the monotone circuit value problem (see, e.g., [34]). The proof essentially follows from the following lemma.

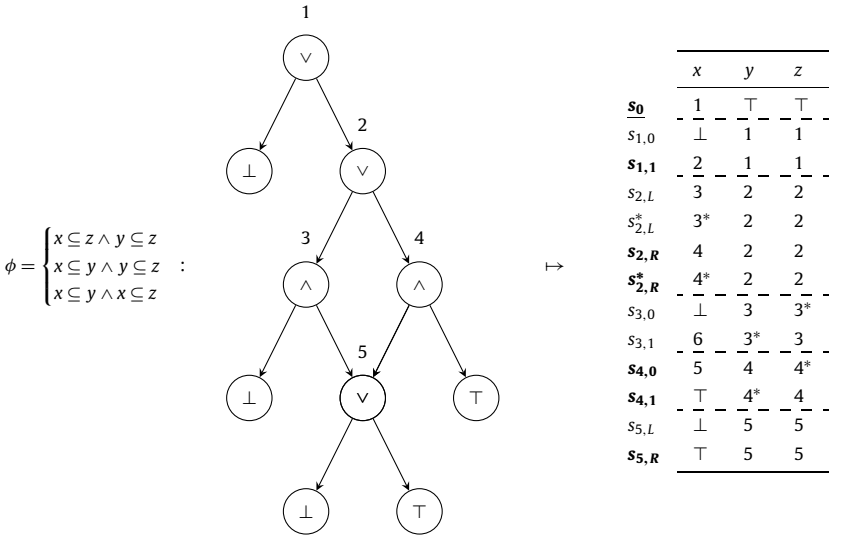
Lemma 22. $\mathsf{MSM}(x \subseteq z \wedge y \subseteq z)$, $\mathsf{MSM}(x \subseteq y \wedge y \subseteq z)$, *and* $\mathsf{MSM}(x \subseteq y \wedge x \subseteq z)$ *are* **P***-complete.*

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



	x	y
$\underline{\boldsymbol{s_{a,b}}}$	a	b
$\boldsymbol{s_{0,a}}$	0	a
$s_{1,0}$	1	0
$s_{2,0}$	2	0
$\boldsymbol{s_{b,0}}$	b	0
$s_{2,1}$	2	1

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



	x	y	z
$\underline{\boldsymbol{s_0}}$	1	$\top$	$\top$
$s_{1,0}$	$\bot$	1	1
$\boldsymbol{s_{1,1}}$	2	1	1
$s_{2,L}$	3	2	2
$s^*_{2,L}$	3^*	2	2
$\boldsymbol{s_{2,R}}$	4	2	2
$\boldsymbol{s^*_{2,R}}$	4^*	2	2
$s_{3,0}$	$\bot$	3	3^*
$s_{3,1}$	6	3^*	3
$\boldsymbol{s_{4,0}}$	5	4	4^*
$\boldsymbol{s_{4,1}}$	$\top$	4^*	4
$s_{5,L}$	$\bot$	5	5
$\boldsymbol{s_{5,R}}$	$\top$	5	5

Fig. 2. MCVP and MSM(ϕ).

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; for this, observe that any gate with indegree l can be replaced with a subcircuit that contains $l-1$ gates with indegree 2. Further, we may assume that the output gate is an OR gate, and there are no consecutive AND gates in the circuit. We annotate each input node of the circuit by its corresponding input $\top$ or $\bot$, and each gate by some distinct number $i \in \mathbb{N} \setminus \{0\}$. Then each gate i takes two input gates i_L, i_R that are either natural numbers or from $\{\top, \bot\}$. Next we construct a team X whose values consist of node annotations $i, \top, \bot$ and distinct copies i^* of AND gates i (see Fig. 2). Define $X := X_{\text{OUT}} \cup X_{\text{AND}} \cup X_{\text{OR}}$, where

$$
\begin{aligned}
X_{\text{OUT}} :=& \{s_0 \colon (x,y,z) \mapsto (1,\top,\top)\}, \text{ where 1 is the output gate;} \\
X_{\text{AND}} :=& \{s_{i,0} \colon (x,y,z) \mapsto (i_L, i, i^*) \mid i \text{ is AND gate}\} \cup \\
& \{s_{i,1} \colon (x,y,z) \mapsto (i_R, i^*, i) \mid i \text{ is AND gate}\}; \text{ and} \\
X_{\text{OR}} :=& \{s_{i,L} \colon (x,y,z) \mapsto (i_L, i, i) \mid i \text{ is OR gate}\} \cup \\
& \{s_{i,R} \colon (x,y,z) \mapsto (i_R, i, i) \mid i \text{ is OR gate}\} \cup \\
& \{s^*_{i,L} \colon (x,y,z) \mapsto (i^*_L, i, i) \mid i \text{ is OR gate}, i_L \text{ is AND gate}\} \cup \\
& \{s^*_{i,R} \colon (x,y,z) \mapsto (i^*_R, i, i) \mid i \text{ is OR gate}, i_R \text{ is AND gate}\}.
\end{aligned}
$$

The definition of X becomes more understandable through the safety game with respect to s_0 and a formula ϕ of the form, say $x \subseteq y \wedge x \subseteq z$. At start, Player I selects either $x \subseteq y$ or $x \subseteq z$. If the output gate is an AND gate, then $x \subseteq y$ represents the left input gate and $x \subseteq z$ the right input gate. Thus Player II has to select the unique assignment from X_{AND} that corresponds to the input gate selected by Player I. If the output gate is an OR gate, then Player II can freely select an assignment X_{OR} that represents either the left or the right input gate. The same procedure is repeated at each gate. Then, Player II has a winning strategy if and only if C accepts w. Note also that the values with an asterisk make X more symmetric and are needed for the remaining instantiations of ϕ. Further, the assignments in X_{AND} and X_{OR} are distinguished from one another by their right subindex that is either a number or a letter, respectively.

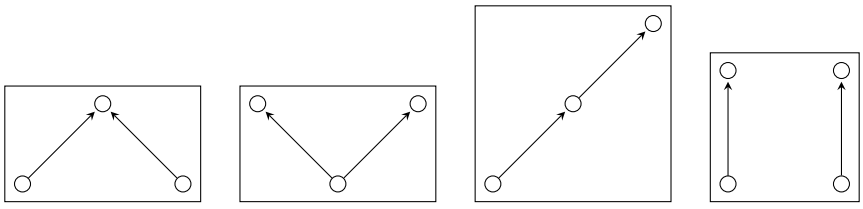


Fig. 3. Subgraphs of G_Σ.

We now show that C accepts w iff $s_0 \in \nu(X, \phi)$. For the only-if direction we actually show a slightly stronger claim: $s_0 \in \nu(X, \phi)$ 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 $Y \subseteq X$ that 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 1 to C'; for each added AND gate i, add both of its input gates; for each added OR gate i, add those input gates that are evaluated true under w. In other words, C' represents a set of paths from the output gate to the input nodes that witnesses the assumption that C accepts w. The team Y will now list the auxiliary values i^* and the gates of C' in each column x, y, z. Define $Y := Y_{\mathrm{OUT}} \cup Y_{\mathrm{AND}} \cup Y_{\mathrm{OR}}$ where

$$Y_{\mathrm{OUT}} := \{s_0\};$$
$$Y_{\mathrm{AND}} := \{s_{i,k} \in X_{\mathrm{AND}} \mid i \in C', k \in \{0, 1\}\}; \text{ and}$$
$$Y_{\mathrm{OR}} := \{s_{i,K} \in X_{\mathrm{OR}} \mid \{i, i_K\} \subseteq C', K \in \{L, R\}\} \cup$$
$$\{s^*_{i,K} \in X_{\mathrm{OR}} \mid \{i, i_K\} \subseteq C', K \in \{L, R\}, i_K \text{ is AND gate}\}.$$

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 (also i^* if i is an AND gate, and $\top$ if $\top$ is an input gate of i), and adding any input gate j of i (also j^* if j is an AND gate) that is 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.

For the converse direction, 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 gate i (OR gate i, resp.) Player I (Player II, resp.) selects the next node from its two input gates 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 $s'(y) = i^*$, which means that further application of $y \subseteq z$ yields s'' from Y with $s''(z) = i^*$ 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. □

Observe that in the reduction $s_0 \in \nu(X, \phi)$ if and only if $\nu(X, \phi)$ is not empty, for any assignment in $\nu(X, \phi)$ propagates a sequence of assignments that ends up with s_0. This observation will be useful later.

The third statement of Theorem 20 now follows. Any G_Σ not covered by Lemma 21 has a subgraph of a form depicted in Fig. 3. Of these G_1–G_3 were considered in Lemma 22, 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. Thus, given G_Σ, fix some subgraph G' of the form G_1–G_4, and construct a team X by Lemma 22. Then, for each node in G_Σ but not in G', append X horizontally by taking a copy of any of its columns. That this suffices follows from the construction in Lemma 22; in particular, from the fact that any true MCVP instance generates a subteam that satisfies all possible unary inclusion atoms between variables x, y, and z. This completes the proof of Theorem 20.

Considering disjunctions, we show now that MSM over a disjunction of non-trivial unary inclusion atoms is **NL**-complete. Note first that membership in **NL** follows Lemma 14. In the proof of **NL**-hardness we start with a disjunction of the form $x \subseteq y \vee \bigvee \Sigma$, where we assume that $y \subseteq x \notin \Sigma$. We use the same reduction from REACH as in Lemma 21 with a simple modification: the assignments $s_{c,d}$ in the team X are now extended to all the variables, other than x and y, occurring in Σ by setting $s_{c,d}(z) = e_z$, where each e_z is a constant such that $e_z \notin V$ and $e_z \neq e_{z'}$ for $z \neq z'$. Clearly this means that $\nu(X, \bigvee \Sigma) = \emptyset$, whence by Lemma 13 $s_{b,a} \in \nu(X, x \subseteq y \vee \bigvee \Sigma)$ if and only if $s_{b,a} \in \nu(X, x \subseteq y)$. As in the proof of Lemma 21, $s_{b,a} \in \nu(X, x \subseteq y)$ is equivalent to b being reachable from a in the input graph G. Thus, $\mathsf{MSM}(x \subseteq y \vee \bigvee \Sigma)$ is indeed **NL**-hard.

For disjunctions of the form $x \subseteq y \vee y \subseteq x \vee \bigvee \Sigma$, we handle the additional variables in Σ in the same way. Then by Lemma 13 $s_{b,a} \in v(X, x \subseteq y \vee y \subseteq x \vee \bigvee \Sigma)$ if and only if $s_{b,a} \in v(X, x \subseteq y)$ or $s_{b,a} \in v(X, y \subseteq x)$. 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 likewise implies that b is reachable from a in G.

In the arguments above, we assumed that all inclusion atoms in Σ were non-trivial. If, on the other hand, Σ contains a trivial inclusion atom $x \subseteq x$, then $s \in v(X, x \subseteq x) \subseteq v(X, \bigvee \Sigma)$ for all $s \in X$, whence $\mathsf{MSM}(\bigvee \Sigma)$ is trivially true for all inputs (X, s).

Corollary 23. *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 general, classifying the complexity of maximal subteam membership for inclusion atoms of arbitrary arity requires different methods. For instance, inclusion atoms of higher arity can describe permutations on variables (consider, e.g., $xyz \subseteq yzx$); this makes their interaction more convoluted than in the case of unary inclusion atoms.

3.4. Teams with a key

In this section we investigate maximal subteam membership for inclusion atoms that correspond to uni-relational foreign keys. That is, we concentrate on inclusion atoms $x \subseteq z$ over teams where the variable z is a key. From the vantage point of databases this restriction is quite natural; in real-world database schemata inclusion dependencies usually only appear as foreign keys to impose referential integrity. What we observe next is that restricting focus to foreign keys seems to also decrease the complexity of maximal subteam membership. We show that the complexity of maximal subteam membership for $x \subseteq z$ collapses from **NL** to **L**, and for $x \subseteq z \wedge y \subseteq z$ it collapses from **P** to **NL**. In Section 4 we will discuss in more detail how maximal subteam membership is connected to database repairing. Here, we only mention that, considering uni-relational foreign key constraints, this problem is roughly equivalent to subset-repair checking over input databases that are consistent with respect to keys.

Definition 24. 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 $s, s' \in X : s(U) = s'(U) \implies s = s'$.

Observe that a key U on a team over V corresponds to the dependence atom $\mathrm{dep}(U, V)$. We now show that maximal subteam for single inclusion atom is **L**-complete if the inclusion atom refers to a key. Given a directed graph $G = (V, E)$ and two vertices a and b, the problem REACH_d is to determine whether G contains a deterministic path from a to b. A *deterministic* path is such that every edge (u, v) in the path is the only edge going out of u. This problem is known to be complete for **L**.

Theorem 25. $\mathsf{MSM}(x \subseteq y)$ *over teams for which y is a key is* **L***-complete.*

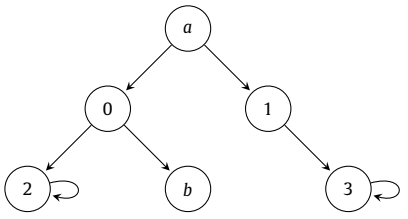
Proof. Hardness. We construct a first-order many-one reduction from REACH_d. Let $G = (V, E)$ be a directed graph, and let a and b be two nodes in the graph. Similarly to Lemma 21, we may assume w.l.o.g. that G is acyclic and that the out-degrees of a and b are respectively 1 and 0. Also, we define E' as the extension of E with (b, a). Then the reduction maps $G' = (V, E')$ to the team

$$X := \{s_{c,d} \mid (d, c) \in E' \wedge \forall (d, c') \in E' : c = c'\},$$

where $s_{u,v}$ is the assignment $(x, y) \mapsto (u, v)$. For example, the graph illustrated in Fig. 1 is mapped to the team that consists of the three assignments $(x, y) \mapsto \{(a, b), (0, a), (2, 1)\}$; observe that the three edges with the same source node 0 are eliminated in the reduction. We show that G contains a deterministic path from a to b iff $s \in v(X, x \subseteq y)$, where s is the assignment that corresponds to the only edge going out from a.

By Lemma 15, $s \in v(X, x \subseteq y)$ iff s is connected to a cycle in the graph $G_X = (X, E)$, where $(s', s'') \in E$ iff $s''(y) = s'(x)$. Since all cycles in G' go through the edge (b, a), accordingly all cycles in G_X go through the assignment $s_{a,b}$. Furthermore, by definition any path in G_X corresponds to a deterministic path in G, and vice versa. Consequently, $s \in v(X, x \subseteq y)$ iff there is a deterministic path from a to b.

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



↦

	x	y	z
$\underline{s_a}$	0	1	a
$\mathbf{s_0}$	2	b	0
s_1	3	3	1
s_2	2	2	2
s_3	3	3	3

Fig. 4. REACH, and MSM($x \subseteq z \wedge y \subseteq z$) over teams with a key z.

Membership. Consider the graph G from Lemma 15 with respect to an assignment s, a team X, and an inclusion atom $x \subseteq y$. Observe that there is a path from a given assignment $s \in X$ to a cycle in G iff some path from s is of length at least $|X|$. Since y is a key on X, all paths in G are deterministic. Thus, the graph condition for $s \in \nu(X, x \subseteq y)$, given in Lemma 15, can be decided 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 26. 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 $G = (V, E)$ 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 nodes have out-degree at most two; to achieve this, it suffices to replace each node with out-degree l by a subgraph that contains $l - 2$ extra nodes.

Since **NL** is closed under complement, we may reduce to the complement of 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; for this, observe that the nodes with out-degree 0 can be recursively removed in logarithmic space.

We define (see also Fig. 4)

$$X := \{s_i \mid i \in V, i \neq b\},$$

where

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

Observe that z is a key on X. By Theorem 17 it suffices to show that Player II has no winning strategy in 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 in 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 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 x 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 MSM($x \subseteq z \wedge y \subseteq z$) over teams with a key z. By Theorem 17 and Lemma 18, $s \notin \nu(X, x \subseteq z \wedge y \subseteq z)$ iff Player II has no winning strategy in safety$_{3 \cdot l}(X, s, x \subseteq z \wedge y \subseteq z)$, 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. □

Observe that the previous theorem can be extended to conjunctions

$$(x_1 \subseteq z \wedge \ldots \wedge x_k \subseteq z),$$

where k is any integer greater than or equal to 2.

Theorem 25 yields immediately the following corollary for disjunction. For the lower bound, MSM($x \subseteq z$) reduces to MSM($x \subseteq z \vee y \subseteq z$) by appending values for y such that $\nu(X, y \subseteq z)$ is empty (see also Lemma 13); for the upper bound, see Lemma 14.

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. 5. Database D.

Corollary 27. MSM($x \subseteq z \vee y \subseteq z$) *over teams for which z is a key, is* **L***-complete.*

4. Consistent query answering

We now turn to the relationship between maximal subteam membership and consistent query answering. The present section is directed to a reader interested in database theory. Reading this section is not necessary for comprehension of subsequent sections, where we turn back to team semantics.

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 practice 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 $\sigma = \{R_1, \ldots, R_n\}$. Each relation symbol 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. Σ) if $D \not\models \Sigma$, and *consistent* otherwise. Given an inconsistent database I, a partial order $\leq$ on databases which share the vocabulary of I, and a set Σ of integrity constraints, a *repair* of 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 $* \in \{\oplus, \text{subset}, \text{superset}\}$ 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. *LAV tgds* (Local As View tuple generating dependencies) are first-order formulae of the form

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

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. For instance,

$$\forall xyz(R(x, y, z) \to \exists u v R(u, x, v))$$

is a unary unirelational inclusion dependency which expresses that any value in the first position of R also appears in the second position of R. Note that unary inclusion atoms over teams correspond to unary inclusion dependencies over unirelational databases.

Example 28. Fig. 5 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 29 *([3]). 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 so-called weakly acyclic collections of LAV tgds enjoy subset-repair checking in logarithmic space [1]. Yet, in general it seems not much attention 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 20 extends to repair checking and consistent query answering. Let C be a complexity class. We say that the $*$-consistent query answering problem is C-complete for Σ if $*$-CQA(Σ, q) is in C for all Boolean conjunctive queries q and C-complete for some such q.

Theorem 30. *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.*

Proof. Subset-repair. Since **NL** and **P** are closed under complement, we may consider the complement of subset-repair checking, that is, the problem of determining whether a database D is not a subset-repair of another database I. Consider first the upper bounds. For (a) Σ is trivially true in D and thus D is a repair if and only if $D = I$; thus clearly the complement of subset-repair is first-order definable. For (b) and (c) observe first that, by the union closure property of inclusion logic (Proposition 5), any database I has a unique subset-repair D_{unique} with respect to Σ; more generally, the same holds for any database I and any set of LAV tgds Σ [3, Corollary 4.3]. Consider two claims: (i) $D \subseteq D_{\text{unique}}$ and (ii) $D \supseteq D_{\text{unique}}$. Since D_{unique} is the union of all $D' \subseteq I$ satisfying Σ, (i) is tantamount to $D \models \Sigma$ and (ii) to $I \setminus D \cap D_{\text{unique}} = \emptyset$. Consequently, with respect to the complement of subset-repair checking, the upper bounds hold for (b) and (c). The lower bounds for (b) and (c) follow because in the reductions of Lemmata 21 and 22 we have that $s_0 \in \nu(X, \phi)$ if and only if $\nu(X, \phi) \neq \emptyset$.

Subset-consistent query answering. 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). Consider first the upper bounds. First, in (a) all inclusion atoms are trivial and thus q itself may be used for the first-order definition. Second, in (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. That is, subset-CQA(Σ, q) is **NL**-hard for $q = R(a, b)$ in (b), and **P**-hard for $q = R(1, \top, \top)$ in (c), where $a, b, 1, \top$ are constant values from the reductions of Lemmata 21 and 22.

$\oplus$-consistent query answering. That the same 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 [3]. 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. Model checking

In this section we discuss the model checking problem for quantifier-free inclusion logic formulae. It turns out that the results of Section 3 are now easily adaptable. As above, we herein restrict attention to quantifier-free formulae.

Definition 31. Let $\phi \in \text{FO}(\subseteq)$. Then 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 32. *Let $\alpha, \beta \in \text{FO}(\subseteq)$ quantifier-free formulae containing no function or constant symbols, and let $C \in \{\mathbf{L}, \mathbf{NL}, \mathbf{P}\}$. Assume further that*

(i) $\text{Fr}(\alpha) \cap \text{Fr}(\beta) = \emptyset$,
(ii) $\text{MSM}(\alpha)$ is C-hard, and
(iii) There is a model $\mathfrak{B}$ and a team Y over $\mathfrak{B}$ with domain $\text{Fr}(\beta)$ such that $\emptyset \neq \nu(\mathfrak{B}, Y, \beta) \subsetneq Y$.

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 $s \in X$. Further, let $\mathfrak{B}$ be a model and Y a team over $\mathfrak{B}$ with domain $\text{Fr}(\beta)$ such that $\emptyset \neq \nu(\mathfrak{B}, Y, \beta) \subsetneq Y$. W.l.o.g. we can assume that $\mathfrak{A}$ and $\mathfrak{B}$ are of the same relational vocabulary and $\text{dom}(\mathfrak{A}) \cap \text{dom}(\mathfrak{B}) = \emptyset$. Indeed, $\mathfrak{A}$ and $\mathfrak{B}$ can be expanded to a common vocabulary by using the empty relation as an interpretation of the added relation symbols. Thus, we can define the (disjoint) union $\mathfrak{A} \cup \mathfrak{B}$ of $\mathfrak{A}$ and $\mathfrak{B}$ in the usual way: $\text{dom}(\mathfrak{A} \cup \mathfrak{B}) = \text{dom}(\mathfrak{A}) \cup \text{dom}(\mathfrak{B})$, and the interpretation $R^{\mathfrak{A} \cup \mathfrak{B}}$ of each relation symbol

is the union of $R^{\mathfrak{A}}$ and $R^{\mathfrak{B}}$. We define a reduction from $(\mathfrak{A}, X, s)$ to a team X' over $\mathfrak{A} \cup \mathfrak{B}$ over $\mathsf{Fr}(\alpha) \cup \mathsf{Fr}(\beta)$ such that $s \in \nu(\mathfrak{A}, X, \alpha)$ iff $\mathfrak{A} \cup \mathfrak{B} \models_{X'} \alpha \vee \beta$.

Let $Z_0 := \nu(\mathfrak{B}, Y, \beta)$ 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

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

By Locality (Proposition 4), we have $\nu(\mathfrak{A} \cup \mathfrak{B}, X', \alpha) \upharpoonright \mathsf{Fr}(\alpha) = \nu(\mathfrak{A} \cup \mathfrak{B}, X' \upharpoonright \mathsf{Fr}(\alpha), \alpha) = \nu(\mathfrak{A}, X, \alpha)$, and similarly $\nu(\mathfrak{A} \cup \mathfrak{B}, X', \beta) \upharpoonright \mathsf{Fr}(\beta) = \nu(\mathfrak{B}, Y, \beta) = Z_0$. Hence, it follows from Lemma 13 that $\mathfrak{A} \cup \mathfrak{B} \models_{X'} \alpha \vee \beta$ iff $t \in \nu(\mathfrak{A}, X, \alpha)$ or $t' \in \nu(\mathfrak{B}, Y, \beta)$ for all $t \in X$ and $t' \in Y$ such that $t \cup t' \in X'$. Moreover, the latter condition holds iff $s \in \nu(\mathfrak{A}, X, \alpha)$.

Finally note that the reduction $(\mathfrak{A}, X, s) \mapsto (\mathfrak{A} \cup \mathfrak{B}, X')$ we defined is of very low complexity: it consists of adding the constant-size model $\mathfrak{B}$ and teams Z_0, Z_1, and defining X' from X, Z_0 and Z_1 with a first-order formula. □

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 in 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 34(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 33. *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 14, 32, 33, Theorem 7, and the results of Sections 3.3 and 3.4 the computational complexity of model checking for various quantifier-free inclusion formulae directly follows. The following proposition covers some example cases. A class $\mathcal{C}$ of pairs of models and teams $(\mathfrak{A}, X)$ is *first-order definable* if there is a first-order sentence ϕ such that

$$(\mathfrak{A}, X) \in \mathcal{C} \iff (\mathfrak{A}, \mathrm{rel}(X)) \models \phi.$$

Recall that $\mathrm{rel}(X) := \{(a_1, \ldots, a_n) \in \mathsf{Dom}(\mathfrak{A})^n \mid s(x_1, \ldots, x_n) = (a_1, \ldots, a_n), s \in X\}$ is the relational encoding of X with domain $\{x_1, \ldots, x_n\}$. Note that the semantics of the inclusion atom is clearly first-order definable, and the same applies to any conjunction of inclusion atoms.

Proposition 34.

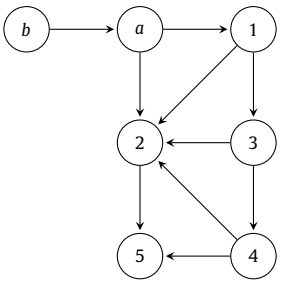
(a) $\mathsf{MC}(x \subseteq y)$ *and* $\mathsf{MC}(x \subseteq y \wedge u \subseteq v)$ *are first-order definable.*
(b) $\mathsf{MC}(x \subseteq y \vee u \subseteq v)$ *and* $\mathsf{MC}(x \subseteq y \vee u = v)$ *are* **NL***-complete.*
(c) $\mathsf{MC}((x \subseteq z \wedge y \subseteq z) \vee u \subseteq v)$ *and* $\mathsf{MC}((x \subseteq z \wedge y \subseteq z) \vee u = v)$ *are* **P***-complete.*
(d) $\mathsf{MC}(x \subseteq y \vee u \subseteq v)$ *over teams for which both y and v are keys, and* $\mathsf{MC}(x \subseteq y \vee u = v)$ *over teams for which y is a key, are* **L***-complete.*
(e) $\mathsf{MC}((x \subseteq z \wedge y \subseteq z) \vee u \subseteq v)$ *and* $\mathsf{MC}((x \subseteq z \wedge y \subseteq z) \vee u = v)$ *over teams for which z is a key are* **NL***-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 $\mathrm{FO}(\subseteq)$. By Lemma 14, we already know that **NL**-computability of maximal subteam membership is preserved under disjunctions. However, Theorem 20 shows that conjunction can increase the complexity of the maximal subteam membership problem from **NL** to **P**-complete, and by Proposition 34, combining a conjunction with a disjunction can lead 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 a well-known **P**-complete problem *Acyclic Geography Game* (AGG) (see, e.g., [16]). An input to AGG 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 AGG if and only if Player II has a winning strategy in $\mathsf{Gm}(V, E, a)$.

The formula we use in the proposition below is $\phi = \forall z(\neg Eyz \vee z \subseteq x)$. Note that given a DAG $G = (V, E)$, a team X and an assignment $s \in X$, $s \in \nu(G, X, \phi)$ if and only if for every node $c \in V$ such that $(s(y), c) \in E$ there is another assignment $s' \in \nu(G, X, \phi)$ such that $s'(x) = c$. This recursive condition is captured by the following game: The positions of the game are assignments in X. In position s, player I chooses $c \in V$ such that $(s(y), c) \in E$. Then player II responds by choosing an assignment $s' \in X$ such that $s'(x) = c$, and the game continues from position s'. The first player unable to move loses.



↦

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

Intuitively any instance of AGG can be encoded as an instance of this game (with the same DAG (V, E)). We will now give a rigorous proof for this intuition.

Proposition 35. *Let ϕ be the formula $\forall z(\neg Eyz \vee z \subseteq x)$. Then* MSM($\phi$) *is* **P***-complete. Consequently,* MC($\phi \vee Euv$) *is also* **P***-complete.*

Proof. We give now a reduction from AGG to MSM(ϕ). Let (V, E, a) be an input to AGG. W.l.o.g. 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 AGG 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. 6). 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 32, we see that 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 universal quantification cannot be freely used if the goal is to construct a fragment of inclusion logic that captures **NL**. In this respect, universal quantification is similar to conjunction. 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 36. *Let $\phi \in \mathrm{FO}(\subseteq)$, $\psi \in \mathrm{FO}$, and let X be a team of a model $\mathfrak{A}$. Then*

(a) $\nu(\mathfrak{A}, X, \exists x\phi) = \{s \in X \mid \exists a \in A : s(a/x) \in X'\}$, *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 $F : X \to \mathcal{P}(A) \setminus \{\emptyset\}$ 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 $X'' = \nu(\mathfrak{A}, X', \phi)$. 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 $X'' \subseteq \nu(\mathfrak{A}, X, \phi \wedge \psi)$. □

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

Proposition 37. *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 36(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 36(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 $\psi \in \mathrm{FO}$, the team X' can be computed in C, whence the claim follows from the assumption that $\mathsf{MSM}(\phi)$ is in C. □

Summarising Lemma 14 and Proposition 37, **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 21, 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 38. $\mathsf{MC}(\phi)$ *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 33. □

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

Proof. By Theorem 9 we may assume w.l.o.g. 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-tuples 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:

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

where $\overline{t}_x$ and $\overline{t}_y$ are n-tuples of fresh variables and

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

As we will soon see, the idea of the tuple $\overline{t}_x$ is to act as a counter which indicates an upper bound for the α-path distance of $\overline{x}$ from $\overline{\min}$.

Assume first 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 $(\overline{x}, \overline{y}, \overline{t}_x, \overline{t}_y)$ 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 $\mathfrak{A} \models \phi'$.

Assume then that $\mathfrak{A} \models \phi'$. Then there is 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 $\overline{t}_x$ 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 $s'(\overline{t}_x) = \overline{\max}$. Lastly, by $\mathfrak{A} \models_X \psi_4$ it follows that $s'(\overline{x}) = \overline{\max}$ which shows that $[\mathrm{TC}_{\overline{x},\overline{y}}\alpha(\overline{x},\overline{y})](\overline{\min},\overline{\max})$. □

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

Theorem 40. *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 also presented a safety game for the maximal subteam membership 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 maximal subteam membership and model checking for quantifier-free inclusion logic formulae that involve inclusion atoms of higher arity and both disjunctions and conjunctions.

In the final part of the article, we turned attention from quantifier-free formulae to formulae with quantifiers. We presented a simple universally quantified formula with only one inclusion atom 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.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

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

References

[1] F.N. Afrati, P.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, 2009, pp. 31–41.
[2] M. Arenas, L.E. Bertossi, J. 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, 1999, pp. 68–79.
[3] B. ten Cate, G. Fontaine, P.G. Kolaitis, On the data complexity of consistent query answering, Theory Comput. Syst. 57 (2015) 843–891, https://doi.org/10.1007/s00224-014-9586-0.
[4] J. Chomicki, J. Marcinkowski, Minimal-change integrity maintenance using tuple deletions, Inf. Comput. 197 (2005) 90–121, https://doi.org/10.1016/j.ic.2004.04.007.
[5] J. Corander, A. Hyttinen, J. Kontinen, J. Pensar, J. 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, 2016, pp. 165–182.
[6] A. Durand, M. Hannula, J. Kontinen, A. Meier, J. Virtema, Approximation and dependence via multiteam semantics, Ann. Math. Artif. Intell. (2018), https://doi.org/10.1007/s10472-017-9568-4.
[7] A. Durand, J. Kontinen, N. de Rugy-Altherre, J. 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, 2015, pp. 73–85.
[8] J. Ebbing, L. Hella, A. Meier, J. Müller, J. Virtema, H. Vollmer, Extended modal dependence logic, in: Logic, Language, Information, and Computation - 20th International Workshop, WoLLIC 2013, Darmstadt, Germany, August 20–23, 2013. Proceedings, 2013, pp. 126–137.
[9] J. Ebbing, J. Kontinen, J. Müller, H. Vollmer, A fragment of dependence logic capturing polynomial time, Log. Methods Comput. Sci. 10 (2014), https://doi.org/10.2168/LMCS-10(3:3)2014.
[10] H. Ebbinghaus, J. Flum, Finite Model Theory, Perspectives in Mathematical Logic, Springer, 1995.
[11] P. Galliani, Inclusion and exclusion dependencies in team semantics: on some logics of imperfect information, Ann. Pure Appl. Log. 163 (2012) 68–84.
[12] P. Galliani, M. Hannula, J. Kontinen, Hierarchies in independence logic, in: Computer Science Logic 2013 (CSL 2013), 2013, pp. 263–280.
[13] P. Galliani, L. Hella, Inclusion logic and fixed point logic, in: S.R.D. Rocca (Ed.), Computer Science Logic 2013 (CSL 2013), Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 2013, pp. 281–295.
[14] E. Grädel, Model-checking games for logics of imperfect information, Theor. Comput. Sci. 493 (2013) 2–14, https://doi.org/10.1016/j.tcs.2012.10.033.
[15] E. Grädel, Games for inclusion logic and fixed-point logic, in: Dependence Logic, Theory and Applications, 2016, pp. 73–98.
[16] R. Greenlaw, H.J. Hoover, W.L. Ruzzo, Limits to Parallel Computation: P-Completeness Theory, Oxford University Press, Inc., USA, 1995.
[17] M. Hannula, L. Hella, Complexity thresholds in inclusion logic, in: R. Iemhoff, M. Moortgat, R.J.G.B. de Queiroz (Eds.), Logic, Language, Information, and Computation - 26th International Workshop, WoLLIC 2019, Utrecht, the Netherlands, July 2–5, 2019, Proceedings, Springer, 2019, pp. 301–322.
[18] M. Hannula, J. Kontinen, A finite axiomatization of conditional independence and inclusion dependencies, Inf. Comput. 249 (2016) 121–137, https://doi.org/10.1016/j.ic.2016.04.001.
[19] M. Hannula, J. Kontinen, J. Virtema, Polyteam semantics, in: Logical Foundations of Computer Science - International Symposium, LFCS 2018, Deerfield Beach, FL, USA, January 8-11, 2018, Proceedings, 2018, pp. 190–210.

[20] M. Hannula, J. Kontinen, J. Virtema, H. Vollmer, Complexity of propositional logics in team semantic, ACM Trans. Comput. Log. 19 (2018) 2:1–2:14, https://doi.org/10.1145/3157054.
[21] L. Hella, A. Kuusisto, A. Meier, H. Vollmer, Satisfiability of modal inclusion logic: lax and strict semantics, ACM Trans. Comput. Log. 21 (2020) 7:1–7:18.
[22] W. Hodges, Compositional semantics for a language of imperfect information, Log. J. IGPL 5 (4) (1997) 539–563.
[23] N. Immerman, Relational queries computable in polynomial time, Inf. Control 68 (1986) 86–104.
[24] N. Immerman, Languages that capture complexity classes, SIAM J. Comput. 16 (1987) 760–778.
[25] N. Immerman, Nondeterministic space is closed under complementation, SIAM J. Comput. 17 (1988) 935–938, https://doi.org/10.1137/0217058.
[26] J. Kontinen, Coherence and computational complexity of quantifier-free dependence logic formulas, Stud. Log. 101 (2013) 267–291, https://doi.org/10.1007/s11225-013-9481-8.
[27] J. Kontinen, A. Kuusisto, P. Lohmann, J. Virtema, Complexity of two-variable dependence logic and if-logic, Inf. Comput. 239 (2014) 237–253, https://doi.org/10.1016/j.ic.2014.08.004.
[28] P. Koutris, J. Wijsen, Consistent query answering for primary keys in logspace, in: 22nd International Conference on Database Theory, ICDT 2019, March 26-28, 2019, Lisbon, Portugal, 2019, pp. 23:1–23:19.
[29] E. Pacuit, F. Yang, Dependence and Independence in Social Choice: Arrow's Theorem, Springer International Publishing, Cham, 2016, pp. 235–260.
[30] J. Väänänen, Dependence Logic, Cambridge University Press, 2007.
[31] J. Väänänen, Modal dependence logic, in: K.R. Apt, R. van Rooij (Eds.), New Perspectives on Games and Interaction, Amsterdam University Press, Amsterdam, 2008.
[32] M.Y. Vardi, The complexity of relational query languages, in: Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing, ACM, 1982, pp. 137–146.
[33] J. Virtema, Complexity of validity for propositional dependence logics, Inf. Comput. 253 (2017) 224–236, https://doi.org/10.1016/j.ic.2016.07.008.
[34] H. Vollmer, Introduction to Circuit Complexity - a Uniform Approach, Texts in Theoretical Computer Science. An EATCS Series, Springer, 1999.
[35] F. Yang, J. Väänänen, Propositional logics of dependence, Ann. Pure Appl. Log. 167 (2016) 557–589, https://doi.org/10.1016/j.apal.2016.03.003.