Disjunctions of Two Dependence Atoms

Nicolas Fröhlich
Leibniz Universität Hannover, Germany

Phokion G. Kolaitis
University of California Santa Cruz, CA, USA
IBM Research, San Jose, CA, USA

Arne Meier
Leibniz Universität Hannover, Germany

Abstract

Dependence logic is a formalism that augments the syntax of first-order logic with dependence atoms asserting that the value of a variable is determined by the values of some other variables, i.e., dependence atoms express functional dependencies in relational databases. On finite structures, dependence logic captures NP, hence there are sentences of dependence logic whose model-checking problem is NP-complete. In fact, it is known that there are disjunctions of three dependence atoms whose model-checking problem is NP-complete. Motivated from considerations in database theory, we study the model-checking problem for disjunctions of two unary dependence atoms and establish a trichotomy theorem, namely, for every such formula, one of the following is true for the model-checking problem: (i) it is NL-complete; (ii) it is L-complete; (iii) it is first-order definable (hence, in AC^0). Furthermore, we classify the complexity of the model-checking problem for disjunctions of two arbitrary dependence atoms, and also characterize when such a disjunction is coherent, i.e., when it satisfies a certain small-model property. Along the way, we identify a new class of 2CNF-formulas whose satisfiability problem is L-complete.

2012 ACM Subject Classification Theory of computation → Problems, reductions and completeness; Theory of computation → Logic; Theory of computation → Logic and databases

Keywords and phrases Dependence logic, coherence, model-checking, complexity, functional dependencies

Digital Object Identifier 10.4230/LIPIcs.CSL.2026.10

Related Version *Full Version*: `https://arxiv.org/abs/2508.16146` [6]

Funding *Nicolas Fröhlich*: The author appreciates funding by the German Research Agency (DFG) under the grant ME2479/3-1 and project id 511769688.
Arne Meier: The author appreciates funding by the German Research Agency (DFG) under the grant ME2479/3-1 and project id 511769688.

Acknowledgements We thank the anonymous reviewers for their valuable comments.

1 Introduction

Notions of dependence and independence are ubiquitous in several different areas of mathematics and computer science, including linear algebra, probability theory, and database theory. Dependence logic $\mathcal{D}$, developed by Väänänen in [14], is a formalism for expressing and studying such notions. The basic building blocks of dependence logic are the *dependence atoms*, which, in effect, express functional dependencies on database relations. Recall that functional dependencies are integrity constraints on databases asserting that the values of some attribute of a database relation is determined by the values of some of the other attributes of that relation (see [1]). For example, if $R(\text{course}, \text{dpt}, \text{inst})$ is a database relation with attributes for courses, departments, and instructors, then the functional dependency $(\text{course} \to \text{dpt})$ asserts that each course is offered by only one department, while the functional dependency (inst → course) asserts that each instructor teaches only one course. In dependence logic $\mathcal{D}$, these two functional dependencies are captured by the dependence atoms $\mathrm{dep}(\mathrm{course}, \mathrm{inst})$ and $\mathrm{dep}(\mathrm{inst}, \mathrm{course})$, respectively. The formulas of $\mathcal{D}$ are obtained by combining dependence atoms, atomic formulas, and negated atomic formulas using conjunction $\wedge$, disjunction $\vee$, existential quantification $\exists$, and universal quantification $\forall$.

Even though the syntax of $\mathcal{D}$ resembles that of first-order logic FO, the semantics of $\mathcal{D}$ is second-order. Instead of single assignments of values to variables, the semantics of $\mathcal{D}$ uses sets of assignments, called *teams*, that represent database relations on which the dependence atoms are interpreted. The second-order character of the semantics of $\mathcal{D}$ is already manifested in the semantics of disjunction: a structure $\mathcal{M}$ and a team T satisfy an $\mathcal{D}$-formula of the form $\varphi_1 \vee \varphi_2$ (in symbols, $(\mathcal{M}, T) \models \varphi_1 \vee \varphi_2$) if there are two teams T_1 and T_2 such that $T = T_1 \cup T_2$, $(\mathcal{M}, T_1) \models \varphi_1$, and $(\mathcal{M}, T_2) \models \varphi_2$. In terms of expressive power and as regards sentences, $\mathcal{D}$ is known to have the same expressive power as existential second-order logic ESO [10, 14]. From this result and Fagin's Theorem [4], it follows that, on the class of all finite structures, the sentences of dependence logic can express precisely all decision problems in NP. In particular, there are $\mathcal{D}$-sentences that can express NP-complete problems; in other words, the model-checking problem for such sentences is NP-complete.

In view of the preceding state of affairs, it is natural to ask: are there syntactic restrictions such that the model-checking problem for $\mathcal{D}$-formulas obeying these restrictions is tractable? Rather surprisingly, Kontinen [8] showed that the model-checking problem can be NP-complete even for disjunctions of three dependence atoms. Concretely, consider the $\mathcal{D}$-formula $\mathrm{dep}(x, y) \vee \mathrm{dep}(u, v) \vee \mathrm{dep}(u, v)$. In [8], it is shown that the following problem is NP-complete: given a team T of arity 4, does $T \models \mathrm{dep}(x, y) \vee \mathrm{dep}(u, v) \vee \mathrm{dep}(u, v)$? In other words, are there three teams T_1, T_2, T_3 such that $T = T_1 \cup T_2 \cup T_3$, $T_1 \models \mathrm{dep}(x, y)$, $T_2 \models \mathrm{dep}(u, v)$, and $T_3 \models \mathrm{dep}(u, v)$? On the tractability side, Kontinen [8] showed that the model-checking problem for disjunctions of two dependence atoms is always in NL, i.e., it is always solvable in non-deterministic logarithmic space. This was achieved via a logspace reduction of the model-checking problem for a disjunction of two dependence atoms to 2SAT, the satisfiability problem for 2-CNF formulas, which is well known to be in NL.

Contributions. In this paper, we carry out a systematic investigation of the disjunctions of two dependence atoms. The motivation for this investigation is twofold. First, while the model-checking for disjunctions of two dependence atoms is always in NL, there are such disjunctions for which the model-checking is NL-complete, while for others it is FO-definable (hence also in uniform AC^0). So, one main aim is to pinpoint the exact complexity of the model-checking problem for a given disjunction of two dependence atoms. Second, disjunctions of dependence atoms capture natural database integrity constraints that have not been considered earlier. To make this point, let us consider the disjunction $\mathrm{dep}(\mathrm{course}, \mathrm{dpt}) \vee \mathrm{dep}(\mathrm{inst}, \mathrm{dpt})$. The disjunct $\mathrm{dep}(\mathrm{course}, \mathrm{dpt})$ expresses the functional dependency that each course is offered by only one department, while the disjunct $\mathrm{dep}(\mathrm{inst}, \mathrm{dpt})$ expresses the functional dependence that each instructor is affiliated with only one department. It is unlikely that either of these two functional dependencies holds in any university, since it is typically the case that some courses are cross-listed by two departments and some instructors have joint appointments in two or more departments. Now, the disjunction $\mathrm{dep}(\mathrm{course}, \mathrm{dpt}) \vee \mathrm{dep}(\mathrm{inst}, \mathrm{dpt})$ expresses the constraint that the database relation $R(\mathrm{course}, \mathrm{dpt}, \mathrm{inst})$ can be split into two parts such that the functional dependency $\mathrm{dep}(\mathrm{course}, \mathrm{dpt})$ holds in the first part and the functional dependency $\mathrm{dep}(\mathrm{inst}, \mathrm{dpt})$ holds in the second part. This is a more relaxed integrity constraint, and it is quite plausible that it holds in many universities. Yet, this type of integrity constraint has not been investigated in database theory. The reason is that the largest collection of integrity constraints in databases studied in the past is the collection of *embedded implicational dependencies* [5], each of which is FO-definable, while, as our result will imply, the disjunction $\mathrm{dep}(\mathrm{course}, \mathrm{dpt}) \vee \mathrm{dep}(\mathrm{inst}, \mathrm{dpt})$ is not FO-definable. We believe that disjunctions of dependence atoms deserve to be studied as database integrity constraints in their own right; the work reported here makes a first step in this direction by focusing on the model-checking problem for such formulas.

We classify the complexity of the model-checking problem for disjunctions of two dependence atoms by establishing a trichotomy theorem: for every $\mathcal{D}$-formula φ that is the disjunction of two dependence atoms, one of the following three statements holds: (i) the model-checking problem for φ is NL-complete; (ii) the model-checking problem for φ is L-complete, where L is the class of all decision problems solvable in deterministic logarithmic space; (iii) the model-checking problem for φ is FO-definable, hence in uniform AC^0, where AC^0 is the class of all decision problems solvable by constant-depth, polynomial-size circuits.

As an illustration of this trichotomy theorem, consider again the database relation $R(\mathrm{course}, \mathrm{dpt}, \mathrm{inst})$ with data about courses, departments, and instructors. Consider also the dependence atoms $\mathrm{dep}(\mathrm{course}, \mathrm{inst})$, $\mathrm{dep}(\mathrm{course}, \mathrm{dpt})$, $\mathrm{dep}(\mathrm{inst}, \mathrm{dpt})$, $\mathrm{dep}(\mathrm{inst}, \mathrm{course})$ that express, respectively, the functional dependencies ($\mathrm{course} \to \mathrm{inst}$), ($\mathrm{course} \to \mathrm{dpt}$), ($\mathrm{inst} \to \mathrm{dpt}$), ($\mathrm{inst} \to \mathrm{course}$). Our trichotomy theorem classifies the complexity of the model-checking for all possible disjunctions of these four dependence atoms. In particular, it implies that the following statements are true:

- The model-checking problem for $\mathrm{dep}(\mathrm{course}, \mathrm{dpt}) \vee \mathrm{dep}(\mathrm{inst}, \mathrm{dpt})$ is NL-complete.
- The model-checking problem for $\mathrm{dep}(\mathrm{inst}, \mathrm{course}) \vee \mathrm{dep}(\mathrm{course}, \mathrm{inst})$ is L-complete.
- The model-checking problem for $\mathrm{dep}(\mathrm{course}, \mathrm{dpt}) \vee \mathrm{dep}(\mathrm{course}, \mathrm{inst})$ is FO-definable, hence it is in uniform AC^0.

As a byproduct of the preceding complexity-theoretic classification, we also identify a collection of 2-CNF formulas for which the satisfiability problem is L-complete.

The preceding results are about *unary* dependence atoms $\mathrm{dep}(x, y)$, where x and y are variables. The syntax of dependence logic allows also for *higher-arity* dependence atoms, i.e., for dependence atoms of the form $\mathrm{dep}(x_1, \dots, x_n, y)$ with $n > 1$; such atoms assert that the value of the variable y is determined by the values of the variables $x_1, \dots, x_n$. We leverage out results about the model-checking problem for disjunctions of two unary dependence atoms to establish a dichotomy theorem about the model-checking problem for disjunctions of two higher-arity dependence atoms: this problem is either NL-complete or FO-definable.

We complement these complexity-theoretic classifications of the model-checking problem with a structural classification of disjunctions of two dependence atoms. Specifically, we determine whether a given disjunction of two dependence atoms is a *coherent* or an *incoherent* $\mathcal{D}$-formula. As defined in [8], a $\mathcal{D}$-formula φ is *coherent* if there is an integer k such that a team T satisfies φ if and only if every sub-team T' of T of size k satisfies φ. If no such k exists, then φ is *incoherent*. We show that a disjunction of two dependence atoms is coherent if and only if the model-checking for this formula is FO-definable. Furthermore, for each coherent disjunction of two dependence atoms, we determine its *coherence level*, i.e., the smallest k that establishes the coherence of that disjunction. Our findings for disjunctions of two unary dependence atoms are summarized in Table 1.

The work reported here paves the way for further exploration of the model-checking problem for the quantifier-free fragment of dependence logic and, perhaps more importantly, it paves the way for further interaction between dependence logic and database theory.

Table 1 Complexity and coherence of disjunctions of two unary dependence atoms.

Formula	Coherence	Model-checking
$\mathrm{dep}(x,y) \vee \mathrm{dep}(z,u)$	incoherent (Prop. 8)	NL-complete (Prop. 10)
$\mathrm{dep}(x,z) \vee \mathrm{dep}(y,z)$	incoherent (Thm. 25*)	NL-complete (Thm. 13*)
$\mathrm{dep}(x,y) \vee \mathrm{dep}(y,z)$	incoherent (Thm. 25*)	NL-complete (Thm. 13*)
$\mathrm{dep}(x,y) \vee \mathrm{dep}(y,x)$	incoherent (Thm. 25*)	L-complete (Thm. 16)
$\mathrm{dep}(x,y) \vee \mathrm{dep}(x,y)$	coherent (level 3) (Prop. 8)	FO (Prop. 9)
$\mathrm{dep}(x,y) \vee \mathrm{dep}(x,z)$	coherent (level 4) (Thm. 22)	FO (Prop. 9)

Due to space constraints, full proofs of results marked with a "$\star$" in the body of the paper are given in the Appendix.

2 Preliminaries

This section contains some basic material about dependence logic. For additional material, we refer the reader to the monograph [14].

A *vocabulary* is a tuple $\tau = (R_1, \ldots, R_m)$ of relation symbols, each of which has a specified natural number r_i as its arity. A τ*-structure* is a tuple $\mathcal{M} = (A, R_1^{\mathcal{M}}, \ldots, R_m^{\mathcal{M}})$, where A is a set, called the *domain* of $\mathcal{M}$, and each $R_i^{\mathcal{M}}$ is an r_i-ary relation on A.

Let Var be a countably infinite set of variables. The syntax of dependence logic $\mathcal{D}$ over a vocabulary τ extends the syntax of first-order logic FO in negation normal form with dependence atoms as additional atomic formulas. A *dependence atom of arity n* is an expression of the form $\mathrm{dep}(y_1, \ldots, y_n, z)$, where $y_1, \ldots, y_n, z$ are distinct variables in Var and $n \geq 0$. If $n = 1$, we say that the dependence atom is *unary*. Intuitively, a dependence atom asserts that the value of the variable z depends on the values of the variables $y_1, \ldots, y_n$. In particular, if $n = 0$, we have a dependence atom of the form $\mathrm{dep}(z)$, which asserts that z is a constant value, since it depends on no other variables; such atoms are called *constancy* atoms.

The formulas of dependence logic $\mathcal{D}$ are defined by the following Backus-Naur form:

$$\psi{:=}\, x_k{=}x_\ell \mid R_i(x_1, \ldots, x_{r_i}) \mid \neg R_i(x_1, \ldots, x_{r_i}) \mid \mathrm{dep}(y_1, \ldots, y_n, z) \mid \psi{\wedge}\psi \mid \psi{\vee}\psi \mid \exists x\psi \mid \forall x\psi,$$

where $x_k, x_\ell, x_1, \ldots, x_{r_i}$ are (not necessarily distinct) variables in Var, R_i is a relation symbol in τ of arity $r_i \in \mathbb{N}$, and $y_1, \ldots, y_n, z$ are distinct variables in Var.

Recall that the semantics of FO-formulas are given using a structure $\mathcal{M}$ and an assignment, i.e., a mapping from Var to the domain of $\mathcal{M}$. Due to the presence of dependence atoms, however, the semantics of $\mathcal{D}$-formulas are given using a structure $\mathcal{M}$ and a set of assignments.

▶ **Definition 1.** *Let X be a set of variables (i.e., $X \subseteq \mathrm{Var}$) and let A be a set.*

- *An* assignment with domain X and range A *is a mapping $s \colon X \to A$.*
- *A* team with domain X and range A *is a set T of assignments with domain X and range A. We will write* $\mathrm{Dom}(T)$ *to denote the domain X of the team T. Similar* $\mathrm{Rng}(T)$ *denotes the range A of the team T.*
- *If T is a team with domain X and $Y \subseteq X$, then the* restriction of T on Y *is the team $T \upharpoonright Y = \{ s \upharpoonright Y \mid s \in T \}$, where $s \upharpoonright Y$ is the restriction of the assignment s to Y.*

Before giving the semantics of $\mathcal{D}$-formulas, we need to introduce two operations on teams.

▶ **Definition 2.** *Let* Var *be the set of all variables and let A be a set.*

- *If $s\colon \mathrm{Var} \to A$ is an assignment with domain* Var *and range A, x is a variable, and a is an element of A, then $s^x_a\colon \mathrm{Var} \to A$ is the assignment such that $s^x_a(x) = a$ and $s^x_a(y) = s(y)$, for every variable $y \neq x$.*
- *If T is a team with domain* Var *and range A, $f\colon T \to A$ is a function, and x is a variable, then the* supplement team *T^x_f is the team $\{s^x_{f(s)} \mid s \in T\}$.*
- *If T is a team with domain* Var *and range A, and x is a variable, then the* duplicate team *T^x_A is the team $\{s^x_a \mid x \in \mathrm{Var} \text{ and } a \in A\}$.*

We are now ready to define the semantics of dependence logic $\mathcal{D}$. In doing so, the supplement team will be used to define the semantics of existential quantification, while the duplicate team will be used to define the semantics of universal quantification.

▶ **Definition 3.** *Let $\tau = (R_1, \dots, R_m)$ be a vocabulary and let $\mathcal{M} = (A, R_1^{\mathcal{M}}, \dots, R_m^{\mathcal{M}})$ be a τ-structure. The satisfaction relation $(\mathcal{M}, T) \models \varphi$, where T is a team with domain* Var *and range A, and φ is a formula of dependence logic $\mathcal{D}$, is defined by induction on the construction of φ and simultaneously for all teams with domain* Var *and range A as follows:*

- *$(\mathcal{M}, T) \models x_k = x_l$ if for all $s \in T$, we have $s(x_k) = s(x_l)$;*
- *$(\mathcal{M}, T) \models R_i(x_1, \dots, x_{r_i})$ if for all $s \in T$, we have $(s(x_1), \dots, s(x_{r_i})) \in R_i^{\mathcal{M}}$;*
- *$(\mathcal{M}, T) \models \neg R_i(x_1, \dots, x_{r_i})$ if for all $s \in T$, we have $(s(x_1), \dots, s(x_{r_i})) \notin R_i^{\mathcal{M}}$;*
- *$(\mathcal{M}, T) \models \mathrm{dep}(y_1, \dots, y_n, z)$ if for all $s_1, s_2 \in T$ with $s_1(y_j) = s_2(y_j)$ for $1 \leq j \leq n$, we have that $s_1(z) = s_2(z)$;*
- *$(\mathcal{M}, T) \models \varphi_1 \wedge \varphi_2$ if $(\mathcal{M}, T) \models \varphi_1$ and $(\mathcal{M}, T) \models \varphi_2$;*
- *$(\mathcal{M}, T) \models \varphi_1 \vee \varphi_2$ if there are teams T_1, T_2 with $T = T_1 \cup T_2$ and $(\mathcal{M}, T_i) \models \varphi_i$, $i = 1, 2$;*
- *$(\mathcal{M}, T) \models \exists x \varphi$ if there is a function $f\colon T \to A$ such that $(\mathcal{M}, T^x_f) \models \varphi$, where T^x_f is the supplement team associated with T, x, and f;*
- *$(\mathcal{M}, T) \models \forall x \varphi$ if $(\mathcal{M}, T^x_A) \models \varphi$, where T^x_A is the duplicate team associated with T and x.*

The semantics of disjunction in dependence logic are quite different from those in first-order logic. In particular, unlike first-order logic, a disjunction of the form $\varphi \vee \varphi$ need not be equivalent to φ. The following examples illustrates this state of affairs.

▶ **Example 4.** Consider the $\mathcal{D}$-formulas $\mathrm{dep}(x, y)$ and $\mathrm{dep}(x, y) \vee \mathrm{dep}(x, y)$. Let $\mathcal{M} = (\{0, 1\})$ be a structure over the empty vocabulary and let $T = \{s, t\}$ be a team with two s and t assignments such that $s(x) = 0, s(y) = 0$ and $t(x) = 0, t(y) = 1$. Clearly $(\mathcal{M}, T) \not\models \mathrm{dep}(x, y)$. Nonetheless, $(\mathcal{M}, T) \models \mathrm{dep}(x, y) \vee \mathrm{dep}(x, y)$, since $(\mathcal{M}, \{s\}) \models \mathrm{dep}(x, y)$ and $(\mathcal{M}, \{t\}) \models \mathrm{dep}(x, y)$.

The next proposition, whose proof can be found in [14], states two important properties of dependence logic. In what follows, if φ is a $\mathcal{D}$-formula, then $\mathrm{free}(\varphi)$ is the (finite) set of all free variables of φ.

▶ **Proposition 5** ([14, Lemma 3.27, Prop. 3.10])**.** *Let $\mathcal{M} = (A, R_1^{\mathcal{M}}, \dots, R_m^{\mathcal{M}})$ be a τ-structure and let φ be a $\mathcal{D}$-formula.*

(Locality) *Let X be a set such that $\mathrm{free}(\varphi) \subseteq X \subseteq \mathrm{Var}$. If T_1, T_2 are teams with domain* Var *and range A such that $T_1 \upharpoonright X = T_2 \upharpoonright X$, then $(\mathcal{M}, T_1) \models \varphi$ if and only if $(\mathcal{M}, T_2) \models \varphi$.*

(Downward Closure) *Let P and T be teams with domain* Var *and range A such that $P \subseteq T$. If $(\mathcal{M}, T) \models \varphi$, then also $(\mathcal{M}, P) \models \varphi$.*

The Locality property tells that the satisfaction relation $(\mathcal{M}, T) \models \varphi$ depends only on the restriction of the team T on the set of the free variables of φ. In particular, if ψ is a $\mathcal{D}$-sentence (i.e., a $\mathcal{D}$-formula with no free variables), then $(\mathcal{M}, T) \models \psi$ if and only $\mathcal{M}, \{\emptyset\}) \models \psi$, where $\{\emptyset\}$ is the team consisting of the empty assignment.

Väänänen [14] showed that $\mathcal{D}$-sentences have the same expressive power as the sentences of existential second-order logic ESO. Combined with Fagin's theorem [4], this result implies that, on the class of all finite structures, $\mathcal{D}$-sentences express precisely all NP problems. Kontinen and Väänänen [11] showed that $\mathcal{D}$-formulas (with free variables) have the same expressive power as the *downward closed* formulas of existential second-order logic ESO (see [11] for the precise statement of this result).

In view of the Locality property, from now on we will only consider teams on finite domains, as long as, in each case, the domain of the team considered contains the free variables of the $\mathcal{D}$-formula at hand.

Coherence. In the following, we present the definition of a small-model property for dependence logic.

▶ **Definition 6** (k-coherence, [8, Def. 3.1])**.** *Let $\varphi(x_1, \ldots, x_n)$ be a quantifier-free $\mathcal{D}$-formula. We say that φ is* k-coherent *if for all structures $\mathcal{M}$ and teams T of range $\mathcal{M}$ such that* $\mathrm{free}(\varphi) \subseteq \mathrm{Dom}(T)$*, the following are equivalent:*

(1.) $(\mathcal{M}, T) \models \varphi$.

(2.) *For all k-element sub-teams $S \subseteq T$ it holds that $(\mathcal{M}, S) \models \varphi$.*

Notice that for the above equivalence, the direction "**(1.)** $\Rightarrow$ **(2.)**" is always true due to $\mathcal{D}$ being downwards closed (see Proposition 5).

▶ **Definition 7** (coherence-level, incoherence)**.** *The* coherence-level *of φ is the least natural number $k \in \mathbb{N}$ such that φ is k-coherent; if such a k does not exist, we call φ* incoherent.

Clearly, first-order atomic formulas have a coherence-level of 1 and dependence atoms a coherence-level of 2.

▶ **Proposition 8.** *The following statements are true:*

(1.) *The formula* $\mathrm{dep}(x, y) \vee \mathrm{dep}(x, y)$ *has coherence-level* 3; *[8, Prop. 3.8]*

(2.) *The formula* $\mathrm{dep}(x, y) \vee \mathrm{dep}(z, u)$ *is incoherent.* *[8, Thm. 3.11]*

We will use the shortcut $\mathrm{Rel}(T) \coloneqq \{(s(x_1), \ldots, s(x_n)) \mid s \in T\}$, when the team T is given as a relation. The following proposition shows that every quantifier-free k-coherent formula of dependence logic is first-order definable (and, hence, is in uniform AC^0).

▶ **Proposition 9** ([8, Thm. 4.9])**.** *Suppose $\varphi(x_1, \ldots, x_n) \in \mathcal{D}$ is a quantifier-free k-coherent formula over a vocabulary τ. Then there is a sentence $\varphi^* \in \mathsf{FO}(\tau \cup \{R\})$, where R is n-ary, such that for all τ-structures $\mathcal{M}$ and for all teams T of domain $\{x_1, \ldots, x_n\}$, we have that $(\mathcal{M}, T) \models \varphi(x_1, \ldots, x_n)$ if and only if* $\mathrm{Rel}(T) \models \varphi^*(R)$.

3 Model-checking

The model-checking problem is a fundamental decision problem arising in every logic. Informally, this problem asks if a finite structure satisfies a formula of the logic at hand. Actually, according to the taxonomy introduced by Vardi [15], there are two versions of the model-checking problem, the *combined complexity* version and the *data complexity* version. In the first version, the input consists of a formula and a finite structure, while in the second version the formula is fixed and the input consists of just a finite structure.

Here, we focus on the data complexity version of the model-checking problem for dependence logic $\mathcal{D}$. Specifically, every $\mathcal{D}$-formula φ gives rise to the following decision problem.

Problem:	MC(φ) – the model-checking problem for $\varphi \in \mathcal{D}$
Input:	A finite τ-structure $\mathcal{M} = (A, R_1^{\mathcal{M}}, \ldots, R_m^{\mathcal{M}})$ and a team T with domain a finite set X and range A such that $\text{free}(\varphi) \subseteq X \subseteq \text{Var}$.
Question:	Does $(\mathcal{M}, T) \models \varphi$?

As mentioned in the introduction, the results in [9, 14] imply that MC(φ) is in NP for every $\mathcal{D}$-formula φ, and that there are $\mathcal{D}$-formulas ψ for which MC(ψ) is an NP-complete problem. Furthermore, the intractability of the model-checking problem is even true for some disjunctions of three dependence atoms; for example, $\text{MC}(\text{dep}(x, y) \vee \text{dep}(u, v) \vee \text{dep}(u, v))$ is NP-complete, as shown by Kontinen [8].

Note that if φ is a $\mathcal{D}$-formula in which no relation symbols from the vocabulary τ occur, then the satisfaction relation $(\mathcal{M}, T) \models \varphi$ depends only on the team T and not on the τ-structure $\mathcal{M}$. From this point on, we will focus on disjunctions of two dependence atoms, hence no relation symbols from the vocabulary τ occur in such dependence logic formulas. For this reason, we will drop $\mathcal{M}$ from the satisfaction relation and, instead, write $T \models \varphi$, where φ contains no relation symbols from τ. Thus, if φ is such a $\mathcal{D}$-formula, then the model-checking problem MC(φ) asks: given a team T with domain a finite set X such that $\text{free}(\varphi) \subseteq X \subseteq \text{Var}$, does $T \models \varphi$? The following result yields a sufficient condition for the model-checking problem to be in NL, the class of all decision problems solvable in non-deterministic logarithmic space.

▶ **Proposition 10** ([8, Thm. 4.12])**.** *Suppose φ and ψ are 2-coherent quantifier-free $\mathcal{D}$-formulas with no relation symbols. Then there is a logarithmic-space reduction from the model-checking problem* MC($\varphi \vee \psi$) *to* 2SAT*. Consequently,* MC($\varphi \vee \psi$) *is in* NL*.*

We now describe the reduction of MC($\varphi \vee \psi$) to 2SAT in some detail, since it will be relevant later on (Section 3.2) in the present paper.

Given a team $T = \{s_1, \ldots, s_k\}$, go through all two-element subsets $\{s_i, s_j\}$ of T, and construct a set C of two-variable clauses, as follows:

- If $\{s_i, s_j\} \not\models \varphi$, then $(x_i \vee x_j) \in C$.
- If $\{s_i, s_j\} \not\models \psi$, then $(\neg x_i \vee \neg x_j) \in C$.

Inuitively, these two clauses encode the property that the team $\{s, t\}$ satisfies neither φ nor ψ. Then let Θ_T be the 2CNF-formula $\bigwedge_{\theta \in C} \theta$. It can be shown that $T \models \varphi \vee \psi$ if and only if Θ_T is satisfiable (see [8] for a detailed proof).

Since every dependence atom is a 2-coherent formula, we obtain the following result.

▶ **Corollary 11.** *If φ and ψ are dependence atoms, then* MC($\varphi \vee \psi$) *is in* NL*.*

3.1 NL-Completeness

This section will cover the cases in which the model-checking problem for disjunctions of two unary dependence atoms is complete for nondeterministic logarithmic space.

▶ **Proposition 12** ([8, Thm. 4.14])**.** *The problem* $\text{MC}(\text{dep}(x, y) \vee \text{dep}(z, w))$ *is* NL*-complete.*

We will outline the reduction, constructed in the proof of this result, because the next result (Theorem 13) is based on this construction. The full proof of Proposition 12 can be found in [8]. The intuition of the construction is that the team encodes the clauses of the formula. The split ensures a consistent assignment on the left side, while the right side allows to "buffer" one unsatisfied literal per clause. Consider a 2SAT instance $\theta(p_1, \ldots, p_m) = \bigwedge_{i \in I} E_i$, with $E_i = (A_{i_1} \vee A_{i_2}), i \in I$, where A_{i_1}, A_{i_2} are literals. For each conjunct E_i, create a team $T_{E_i} = \{s_{i_1}, s_{i_2}\}$ with domain $\{x, y, z, w\}$, where variable x ranges over $\{p_1, \ldots, p_m\}$; variable

T_{E_i}	x	y	z	w
s_{i_1}	p_k	1	i	0
s_{i_2}	p_j	0	i	1

T_{E_i}	x	y	z
s_{i_1}	p_k	i	1_k
s_{i_2}	p_j	i	0_j

T_{E_i}	x	y	z
s_{i_1}	i	p_k	1
s_{i_2}	i	p_j	0

Figure 1 Teams for $(p_k \vee \neg p_j)$ in Proposition 12 (left) and Theorem 13 (middle/right).

y ranges over the truth values $\{0, 1\}$; variable z ranges over the indices $i \in I$, thus z denotes the clause E_i; and variable w ranges over the values $\{0, 1\}$. Variables z and w ensure that at least one of the assignments from each T_{E_i} go into the subset of T that eventually encodes an assignment satisfying θ. For example, the team T_{E_i} for a clause $(p_k \vee \neg p_j)$ is the one on the left in Figure 1. Finally $T = \bigcup_{i \in I} T_{E_i}$.

▶ **Theorem 13** ($\star$). *The following statements are true:*
(1.) *The problem* $\mathrm{MC}(\mathrm{dep}(x, z) \vee \mathrm{dep}(y, z))$ *is* NL*-complete.*
(2.) *The problem* $\mathrm{MC}(\mathrm{dep}(x, y) \vee \mathrm{dep}(y, z))$ *is* NL*-complete.*

Proof sketch. Membership follows from Proposition 10. For hardness, we proceed as in the proof of Proposition 12, that is, we transform a 2CNF formula into a team, additionally, we require that the clauses do not contain the same variable twice. For the first case, $\mathrm{dep}(x, z) \vee \mathrm{dep}(y, z)$, we use T_{E_i} shown in the middle of Figure 1. For the second case, $\mathrm{dep}(x, y) \vee \mathrm{dep}(y, z)$, we use T_{E_i} depicted in the right of Figure 1. Furthermore, let again I be the set of clause indices and m be the number of propositions in the 2CNF formula θ.

(1.) Variable x ranges over $\{p_1, \ldots, p_m\}$, variable y ranges over I, and variable z ranges over $\{\, 0_k, 1_k \mid 1 \leq k \leq m \,\}$, that is, each proposition has its own truth values. The dependence atom $\mathrm{dep}(x, z)$ ensures the existence of a consistent assignment that satisfies θ, while $\mathrm{dep}(y, z)$ allows only one of s_{i_1} and s_{i_2} to be on its "side of the split."

(2.) Variable x ranges over I, variables y ranges over $\{p_1, \ldots, p_m\}$, and variable z ranges over $\{0, 1\}$. Here, the atom $\mathrm{dep}(y, z)$ ensures the existence of a consistent assignment that satisfies θ, while $\mathrm{dep}(x, y)$ ensures only one of s_{i_1} and s_{i_2} is on its "side of the split." ◀

3.2 L-Completeness

This section will cover the cases in which the model-checking problem for disjunctions of two unary dependence atoms is complete for logarithmic space.

If T is a team with domain X and x is a variable in X, then we will use the notation $\mathrm{Rng}_x(T) \coloneqq \mathrm{Rng}(T \upharpoonright \{x\})$ to denote the set of values in the range of T that x takes.

▶ **Definition 14.** *Let T be a team with domain $\{x, y\}$ such that* $\mathrm{Rng}_x(T) \cap \mathrm{Rng}_y(T) = \emptyset$. *The* undirected graph *of T is $G_T \coloneqq (V_T, E_T)$, where*

$$V_T = \mathrm{Rng}(T), \qquad E_T = \left\{\, \{s(x), s(y)\} \,\middle|\, s \in T \,\right\}.$$

Notice that, by definition, G_T is a bipartite graph. The next lemma provides a graph-based interpretation of the condition under which a team T satisfies the formula $\mathrm{dep}(x, y) \vee \mathrm{dep}(y, x)$.

▶ **Lemma 15** ($\star$). *Let T be a team with domain $\{x, y\}$ such that* $\mathrm{Rng}_x(T) \cap \mathrm{Rng}_y(T) = \emptyset$. *The following statements are equivalent:*
(1.) $T \models \mathrm{dep}(x, y) \vee \mathrm{dep}(y, x)$.
(2.) *G_T can be transformed into a directed graph $G'_T = (V_T, E'_T)$ such that*
  (a) *each node has at most one out-going edge, and*
  (b) *for each $\{u, v\} \in E_T$ either $(u, v) \in E'_T$ or $(v, u) \in E'_T$.*
(3.) *For each connected component $C = (V_C, E_C)$ of G_T, we have that $|E_C| \leq |V_C|$.*

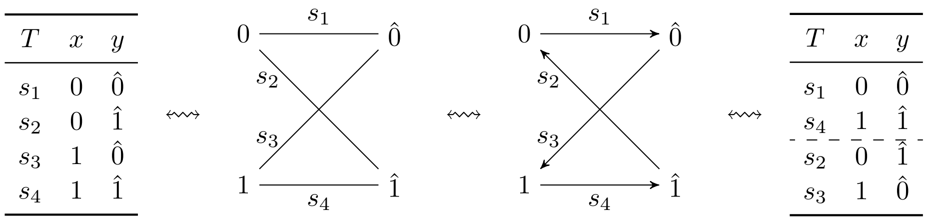


Figure 2 Example depicting the construction of G_T from team T on the left. Followed by the transformation into directed graph G'_T, where each node has at most one out-going edge. This induces the split $\{s_1, s_4\} \models \mathrm{dep}(x, y)$ and $\{s_2, s_3\} \models \mathrm{dep}(y, x)$ of T shown on the right.

Algorithm 1 Model-checking $\mathrm{dep}(x, y) \vee \mathrm{dep}(y, x)$.

input : Graph G_T — `// Space`

foreach $v \in V_T$ **do** — `//` $\log |V|$

  $c \leftarrow 0$ — `//` $\log |V|$

  $d \leftarrow$ number of incident edges of v — `//` $\log |E|$

  foreach $u \in V_T \setminus \{v\}$ *and* u *is reachable by* v **do** — `//` $\log |V|$

    $c \leftarrow c + 1$, $d \leftarrow d +$ number of incident edges of u

  if $\frac{d}{2} > c$ **then** **return** `false` — `//` d `counts the edges twice`

return `true`

A visual example for G_T and the equivalence of **(1.)** and **(2.)** are depicted in Figure 2.

▶ **Theorem 16.** *The problem* $\mathrm{MC}(\mathrm{dep}(x, y) \vee \mathrm{dep}(y, x))$ *is in* L.

Proof. Algorithm 1 decides $T \models \mathrm{dep}(x, y) \vee \mathrm{dep}(y, x)$ requiring only logarithmic space.

We assume the input to the algorithm to be the graph G_T, which is possible since its edge set is, depending on the used encoding, the same as the team T. The correctness follows immediately from Lemma 15, i.e., Algorithm 1 returns `true` if and only if $|E_C| \leq |V_C|$ for all connected components C of G_T. The algorithm iterates over all nodes v in G_T and checks that the size of the corresponding connected component (so its number of nodes) is not smaller than its number of edges. The size count, c, is achieved by counting the number of reachable nodes. In parallel, the sizes of their respective neighborhoods are added up in a different counter, d. Since this counts each edge twice, halving d and comparing to c suffices.

The amount of space required by the algorithm is as follows. Computing the number of nodes in a connected component can be done in logarithmic space in $|G_T|$ (which is in $O(|T|)$) using undirected reachability (which is well-known to be in L by Reingold's theorem [13]). The algorithm stores an index of the current node v and an index of an auxiliary node u, hence both are, by binary encoding, in $O(\log |T|)$. Furthermore, two counters are used: one to count the size of the connected component, c, and another to count the number of edges in the connected component, d. Both counters are, again, in $O(\log |T|)$. Storing these values together requires only logspace in the input size. The claimed result applies, as $\mathrm{L}^{\mathrm{L}} = \mathrm{L}$ (that is to say, L is low for itself). ◀

▶ **Lemma 17** ($\star$). *Let T be a team with domain $\{x, y\}$. If the size $|T| > |\mathrm{Rng}_x(T)| + |\mathrm{Rng}_y(T)|$, then $T \not\models \mathrm{dep}(x, y) \vee \mathrm{dep}(y, x)$.*

In the following, we will utilize the definition of *first-order reductions* as defined by Immerman [7] and Dahlhaus [3].

$T_{\{a,b\}}$	x	y
s_1	a	b
s_2	b	a

$T_{\{\top,u\}}$	x	y
s_1	$\top$	u
s_2	u	$\top$

$T_{\{\top,v\}}$	x	y
s_1	$\top$	v
s_2	v	$\top$

$T_{\{\bot,u\}}$	x	y
s_1	$\bot$	u
s_2	u	$\bot$

$T_{\{\bot,v\}}$	x	y
s_1	$\bot$	v
s_2	v	$\bot$

Figure 3 Team $T_{\{a,b\}}$ for edge $\{a,b\} \in E$ and teams $T_{\{\top,u\}}, T_{\{\top,v\}}, T_{\{\bot,u\}}, T_{\{\bot,v\}}$.

Figure 4 (Left): Example UFA instance. (Right): Visualization of partition into T_1 (solid lines) and T_2 (dashed lines).

▶ **Theorem 18.** *The problem* $\mathrm{MC}(\mathrm{dep}(x,y) \vee \mathrm{dep}(y,x))$ *is* L*-hard under first-order reductions.*

Proof. We reduce from the complement of UFA (which is known to be L-complete [2]):

Problem:	UFA – Undirected Forest Accessibility
Input:	acyclic undirected graph $G = (V, E)$, nodes u and v.
Question:	is there is a path between u and v?

Suppose $\langle G = (V, E), u, v\rangle$ is an instance of UFA. We will construct a team T, such that $T \models \mathrm{dep}(x,y) \vee \mathrm{dep}(y,x)$ if and only if there is *no path* between u and v in G.

For each undirected edge $\{a,b\} \in E$, create a team $T_{\{a,b\}}$ with two assignments s and t, such that $s(x) = t(y) = a$ and $s(y) = t(x) = b$. Next, let $\top$ and $\bot$ be two elements such that $\{\top,\bot\} \cap V = \emptyset$. Create four more teams $T_{\{\top,u\}}, T_{\{\top,v\}}, T_{\{\bot,u\}}$, and $T_{\{\bot,v\}}$ as depicted in Figure 3. Now, T is the union $\bigcup_{\{a,b\}\in E} T_{\{a,b\}} \cup \{T_{\{\top,u\}}, T_{\{\top,v\}}, T_{\{\bot,u\}}, T_{\{\bot,v\}}\}$. The team T can be constructed using an FO query $I\colon \mathrm{STRUC}[\tau_g] \to \mathrm{STRUC}[\tau_{2t}]$, where $\tau_g = (E^2)$ is the vocabulary of graphs with edge relation E and $\tau_{2t} = (T^2)$ is the vocabulary of teams with domain of size two, encoded via the binary relation $\mathrm{Rel}(T)$ (see, Proposition 9).

Now, we will show that $\langle G = (V, E), u, v\rangle \in \overline{\mathrm{UFA}}$ if and only if $T \models \mathrm{dep}(x,y) \vee \mathrm{dep}(y,x)$.

"$\Rightarrow$": Assume that there is no path between u and v in G. Then take u and v as roots for trees corresponding to their acyclic connected components. For all other connected components choose any node as root. We partition T into two teams as follows:

$$\begin{aligned}
T_1 &= \{s \in T \mid s(y) \text{ is the parent node of } s(x)\}\\
&\quad \cup \{s \in T \mid (s(x), s(y)) = (u, \bot)\} \cup \{s \in T \mid (s(x), s(y)) = (v, \top)\}\\
&\quad \cup \{s \in T \mid (s(x), s(y)) = (\bot, v)\} \cup \{s \in T \mid (s(x), s(y)) = (\top, u)\}\\
T_2 &= T \setminus T_1 = \{s \in T \mid (s(x), s(y)) = (t(y), t(x)), t \in T_1\}
\end{aligned}$$

Observe that T_2 contains the "reverse directions" of T_1 (see Figure 4). Now, $s(x) \neq t(x)$ for all $s, t \in T_1$, because each $s(x) \in V$ only has one parent $s(y) \in V$ in their corresponding trees (the cases containing $\top$ or $\bot$ satisfy this condition by construction of T_1, as they appear only once for x; and u, v are roots, so do not have parents themselves). Thus, T_1 trivially satisfies $\mathrm{dep}(x,y)$. Since the construction and definition of T_2 are symmetric, $s(x) \neq t(x)$ for all $s, t \in T_2$. This yields $T_2 \models \mathrm{dep}(y,x)$.

"$\Leftarrow$": For the other direction, assume the contraposition, i.e., that there is a path between u and v in G. Let $C = (V_C, E_C)$ be the connected component with $\{u,v\} \subseteq V_C$. We will focus now on the sub-team for C together with the special assignments (shown in Figure 3) and argue why this sub-team cannot be split to satisfy $\mathrm{dep}(x,y) \vee \mathrm{dep}(y,x)$ (this would imply that the full team cannot be split as well). For that purpose, let T_C be the union $\bigcup_{\{a,b\}\in C} T_{\{a,b\}}$ of the teams created by the edges in C. Observe that

$$|T_C| = 2|E_C| = 2(|V_C| - 1) \text{ and } |\mathrm{Rng}_x(T)| = |\mathrm{Rng}_y(T)| = |V_C|,$$

i.e., the size of the team is twice the number of edges in C, because each edge adds two assignments by definition. The number of edges in C is then one less than the number of vertices in C. The sizes of the domains of x and y are the same as the number of nodes in C.

Next, let $T'_C = T_C \cup \{T_{\{\top,u\}}, T_{\{\top,v\}}, T_{\{\bot,u\}}, T_{\{\bot,v\}}\}$, then

$$|T'_C| = 2|E_C| + 8 = 2(|V_C| - 1) + 8 \text{ and } |\mathrm{Rng}_x(T)| = |\mathrm{Rng}_y(T)| = |V_C| + 2,$$

because eight new assignments are added to the team and two new elements to the domains. Thus, we get

$$|T'_C| = 2(|V_C| - 1) + 8 = 2|V_C| + 6 > 2|V_C| + 4 = |\mathrm{Rng}_x(T)| + |\mathrm{Rng}_y(T)|$$

and, by Lemma 17, we have that $T'_C \not\models \mathrm{dep}(x,y) \vee \mathrm{dep}(y,x)$. Therefore $T \not\models \mathrm{dep}(x,y) \vee \mathrm{dep}(y,x)$ is true, as $T'_C \subseteq T$ and due to downwards closure. ◀

A restricted case of Krom-satisfiability. In the sequel, we will show how our results also establish the L-completeness of the satisfiability problem restricted to a particular type of 2CNF formulas, namely, those that are monotone, transitive, and dual-free.

▶ **Definition 19.** *Let ϕ be a 2CNF formula with variables $\mathrm{Var}(\phi) = \{x_1, \dots, x_n\}$, such that:*

(1.) *Every clause is monotone, i.e., it is of the form $(x_i \vee x_j)$ or of the form $(\bar{x}_i \vee \bar{x}_j)$. The first type is called* positive*, the second type* negative*.*

(2.) *No two "dual" clauses $(x_i \vee y_j)$, $(\bar{x}_i \vee \bar{x}_j)$ appear in ϕ.*

(3.) *Transitivity holds: if $(x_i \vee x_j)$ and $(x_j \vee x_k)$ are clauses of ϕ, then so is $(x_i \vee x_k) \in \phi$; furthermore, if $(\bar{x}_i \vee \bar{x}_j)$ and $(\bar{x}_j \vee \bar{x}_k)$ are clauses of ϕ, then so is $(\bar{x}_i \vee \bar{x}_k) \in \phi$.*

We call such 2CNF formulas monotone transitive dual-free *and abbreviate the corresponding satisfiability problem by* mtdf-2SAT*.*

▶ **Lemma 20** ($\star$)**.** *The problems* $\mathrm{MC}(\mathrm{dep}(x,y) \vee \mathrm{dep}(y,x))$ *and* mtdf-2SAT *are equivalent via first-order reductions.*

Proof sketch. Notice how the 2CNF formulas in the reduction of Proposition 10 are monotone, transitive and dual-free, if we have that $\varphi = \mathrm{dep}(x,y)$ and $\psi = \mathrm{dep}(y,x)$. This gives a one-to-one correspondence, thus the other directions is simply the inverse. ◀

Lemma 20 together with Theorem 16 and Theorem 18 yield the following result.

▶ **Theorem 21.** *Satisfiability of monotone transitive dual-free 2CNF formula is* L*-complete under first-order reductions.*

4 Coherence

This section is devoted to identifying the coherent disjunctions of two unary dependence atoms and determining their coherence-level.

▶ **Theorem 22.** *The formula* $\mathrm{dep}(x,y) \vee \mathrm{dep}(x,z)$ *has coherence-level* 4*.*

$\text{dep}(x, y) \vee \text{dep}(x, z)$

T	x	y	z
s_1	1	1	1
s_2	1	1	2
s_3	1	2	1
s_4	1	2	2

(a)

$\text{dep}(x) \vee \text{dep}(y, z)$

T	x	y	z
s_1	1	1	1
s_2	1	1	2
s_3	2	2	1
s_4	2	2	2

(b)

$\text{dep}(x) \vee \text{dep}(x, y)$

T	x	y
s_1	1	1
s_2	1	2
s_3	2	1
s_4	2	2

(c)

$\text{dep}(y) \vee \text{dep}(x, y)$

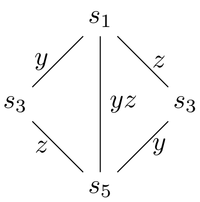
T	x	y
s_1	1	1
s_2	2	1
s_3	3	2
s_4	4	2

(d)

Figure 5 Counterexamples of 3-coherence for formulas used in the proofs of Thms. 22 and 24.

T	x	y	z
s_1	0	0	0
s_2	0	0	1
s_3	0	1	2
s_4	0	2	0
s_5	0	2	1

$\rightsquigarrow$



T	x	y	z
s_1	0	0	0
s_2	0	0	2
s_3	0	1	1
s_4	0	2	1
s_5	1	2	0

$\rightsquigarrow$

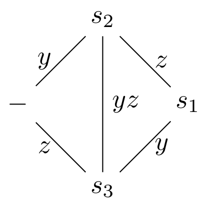


Figure 6 Two examples for the construction in Theorem 22. Edge label denote the variable values where the assignments from the vertices differ, e.g., edge $\{s_1, s_3\}$ with label y means that $s_1(y) \neq s_3(y)$. On the left we have that s_1 corresponds to s, s_3 to u and v, and s_5 to t from the construction.

Proof. We first show that $\text{dep}(x, y) \vee \text{dep}(x, z)$ is not 3-coherent. Consider the team T in Figure 5 (a). This team does not satisfy $\text{dep}(x, y) \vee \text{dep}(x, z)$. To see this, consider the two maximal subsets, $T_1 = \{s_1, s_2\}$ and $T_2 = \{s_3, s_4\}$, that satisfy $\text{dep}(x, y)$. We cannot split $T = T_1 \cup T_2$, because $T_1 \not\models \text{dep}(x, z)$ and $T_2 \not\models \text{dep}(x, z)$.

Every 3-element subset of T satisfies $\text{dep}(x, y) \vee \text{dep}(x, z)$, because either T_1 or T_2 is now a singleton (overlapping splits not need to be considered), thus trivially satisfying the dependence atom $\text{dep}(x, z)$. Therefore $\text{dep}(x, y) \vee \text{dep}(x, z)$ is not 3-coherent.

Next, we show that if T does not satisfies $\text{dep}(x, y) \vee \text{dep}(x, z)$, then there is a sub-team $T' \subseteq T$ of size $|T'| \leq 4$ that does not satisfies $\text{dep}(x, y) \vee \text{dep}(x, z)$ either. Let T be a team with $T \not\models \text{dep}(x, y) \vee \text{dep}(x, z)$. If $|T| \leq 4$, then $T' = T$. Otherwise assume without loss of generality that all assignments in T agree on x. Two assignments with different values for x trivially satisfy both atoms. Thus, we can make this assumption by splitting T into sub-teams that agree on x, and then consider each sub-team separately.

Now, let $T' = \{s, u, v, t\} \subseteq T$, such that the following constraints hold: $s(y) \neq t(y)$ and $s(z) \neq t(z)$; $u(y) \neq s(y)$ and $u(z) \neq t(z)$; $v(z) \neq s(z)$ and $v(y) \neq t(y)$. We first show that $T' \not\models \text{dep}(x, y) \vee \text{dep}(x, z)$ is true and then that such a $T' \subseteq T$ always exists (see Figure 6). By the constraints on y, we have that $\{s, v\}$ and $\{u, t\}$ are the two maximal sub-teams that satisfy $\text{dep}(x, y)$. Therefore at least one of the two sub-teams must also satisfy $\text{dep}(x, z)$ for the disjunction to be true. But $\{s, v\} \not\models \text{dep}(x, z)$, because $s(x) = v(x)$ and $s(z) \neq v(z)$; analogously $\{u, t\} \not\models \text{dep}(x, z)$. Thus $T' \not\models \text{dep}(x, y) \vee \text{dep}(x, z)$ is true.

Let $s \in T$ and assume no $t \in T$ meets the constraints above. Then T can be split via $T_1 = \{w \in T \mid w(y) = s(y)\} \models \text{dep}(x, y)$ and $T_2 = \{w \in T \mid w(z) = s(z)\} \models \text{dep}(x, z)$. Next, assume there are s, t and u, but no v. Then we split T via $T_1 = \{w \in T \mid w(y) = t(y)\} \models \text{dep}(x, y)$ and $T_2 = \{w \in T \mid w(z) = s(z)\} \models \text{dep}(x, z)$. The last case is analogous. This is a contradiction to $T \not\models \text{dep}(x, y) \vee \text{dep}(x, z)$, hence such a sub-team must always exists. ◀

As a special case of the previous lemma (ignore the x column), we deduce the following corollary.

▶ **Corollary 23.** *The formula* $\mathrm{dep}(y) \vee \mathrm{dep}(z)$ *has coherence-level* 4.

▶ **Theorem 24** ($\star$)**.** *Each of the following formulas has coherence-level* 4*:*
(1.) $\mathrm{dep}(x) \vee \mathrm{dep}(y,z)$*;* **(2.)** $\mathrm{dep}(x) \vee \mathrm{dep}(x,y)$*;* **(3.)** $\mathrm{dep}(y) \vee \mathrm{dep}(x,y)$.

Proof sketch. For the lower bound consider the teams (b)–(d) in Figure 5. The upper bound works with a similar construction as in the proof of Theorem 22. ◀

Finally, we turn towards the three incoherent cases. For these formulas, there exists no $k \in \mathbb{N}$, such that the respective formula is k-coherent. Notice that the incoherence result can here be deduced from the fact that $\mathsf{FO} \subsetneq \mathrm{L}$. Nevertheless, we provide independent proofs for these results in the appendix.

▶ **Theorem 25** ($\star$)**.** *The following three formulas are incoherent:*
(1.) $\mathrm{dep}(x,y) \vee \mathrm{dep}(y,x)$*;* **(2.)** $\mathrm{dep}(x,y) \vee \mathrm{dep}(y,z)$*;* **(3.)** $\mathrm{dep}(x,z) \vee \mathrm{dep}(y,z)$.

By combining the results from Section 3 and 4, we obtain the following complete classification for the complexity of the model-checking of disjunctions of two dependence or constancy atoms.

▶ **Theorem 26.** *The following statements are true for the model-checking problem of disjunctions of two unary dependence or constancy atoms:*
(1.) *If* φ *is one of the formulas* $\mathrm{dep}(x,y)\vee\mathrm{dep}(z,u)$, $\mathrm{dep}(x,z)\vee\mathrm{dep}(y,z)$, $\mathrm{dep}(x,y)\vee\mathrm{dep}(y,z)$, *then the problem* $\mathrm{MC}(\varphi)$ *is* NL*-complete.*
(2.) *The problem* $\mathrm{MC}(\mathrm{dep}(x,y) \vee \mathrm{dep}(y,x))$ *is* L*-complete.*
(3.) *In all other cases, the problem* $\mathrm{MC}(\varphi)$ *is in* FO*, hence also in uniform* AC^0.

5 Disjunctions of higher-arity dependence atoms

In this section, we consider disjunctions of two dependence atoms whose arity may be higher than one. We leverage the preceding results about disjunctions of two unary dependence atoms to classify the complexity of disjunctions of two dependence atoms of higher arity.

Before stating and proving the main result of this section, we present two lemmas. The idea in the next lemma is to encode a subset of the domain by a tuple and also change the range appropriately.

▶ **Lemma 27.** *Consider a* $\mathcal{D}$*-formula of the form* $\mathrm{dep}(x_1,\dots,x_n,y)\vee\mathrm{dep}(u_1,\dots,u_m,v)$*. For every team* $T=\{s_1,\dots,s_k\}$ *with domain* $\{x_1,\dots,x_n,y,u_1,\dots,u_m,v\}$ *and range* A*, there is a team* T' *with domain* $\{x',y,u_1,\dots,u_m,v\}$ *and range* $A^n\cup A$*, such that*

$$T \models \mathrm{dep}(x_1,\dots,x_n,y)\vee\mathrm{dep}(u_1,\dots,u_m,v) \quad \textit{iff} \quad T' \models \mathrm{dep}(x',y)\vee\mathrm{dep}(u_1,\dots,u_m,v).$$

Proof. Consider the assignments s_i', $1 \le i \le k$, with domain $\{x',y,u_1,\dots,u_m,y\}$ and range $A^n \cup A$ defined as follows: $s_i'(x') = (s(x_1),\dots,s(x_n))$ and $s_i'(w) = s_i(w)$, if w is one of the variables $y,u_1,\dots,u_m,v$. Furthermore, consider the team $T' = \{s_1',\dots,s_k'\}$ be the team, which has domain $\{x',y,u_1,\dots,u_m,v\}$ and range $A^n\cup A$.

Assume first that $T \models \mathrm{dep}(x_1,\dots,x_n,y)\vee\mathrm{dep}(u_1,\dots,u_m,v)$ via $T = T_1\cup T_2$, that is, $T_1 \models \mathrm{dep}(x_1,\dots,x_n,y)$ and $T_2 \models \mathrm{dep}(u_1,\dots,u_m,v)$. Let $T_1' = \{s_i' \in T' \mid s_i \in T_1\}$ and $T_2' = \{s_i' \in T' \mid s_i \in T_2\}$. Then the following hold:

$$
\begin{aligned}
T_1 \models \mathrm{dep}(x_1,\ldots,x_n,y) &\text{ iff } \forall s,t \in T_1\colon s(x_1)=t(x_1),\ldots,s(x_n)=t(x_n) \Rightarrow s(y)=t(y)\\
&\text{ iff } \forall s,t \in T_1\colon (s(x_1),\ldots,s(x_n))=(t(x_1),\ldots,t(x_n)) \Rightarrow s(y)=t(y)\\
&\text{ iff } \forall s',t' \in T_1'\colon s'(x')=t'(x') \Rightarrow s'(y)=t'(y) \text{ iff } T_1' \models \mathrm{dep}(x',y).
\end{aligned}
$$

Furthermore, $T_2' \models \mathrm{dep}(u_1,\ldots,u_m,v)$, since $T_2 \restriction \{u_1,\ldots,u_m,v\} = T_2' \restriction \{u_1,\ldots,u_m,v\}$. Thus, if $T \models \mathrm{dep}(x_1,\ldots,x_n,y) \vee \mathrm{dep}(u_1,\ldots,u_m,v)$, then $T' \models \mathrm{dep}(x',y) \vee \mathrm{dep}(u_1,\ldots,u_m,v)$. The converse is established using a similar argument. ◀

By applying Lemma 27 twice, we obtain the following result.

▶ **Corollary 28.** *Consider a $\mathcal{D}$-formula of the form $\mathrm{dep}(x_1,\ldots,x_n,y) \vee \mathrm{dep}(u_1,\ldots,u_m,v)$. For every team $T=\{s_1,\ldots,s_k\}$ with domain $\{x_1,\ldots,x_n,y,u_1,\ldots,u_m,v\}$ and range A, there is a team T' with domain $\{x',y,u',v\}$ and range $A^n \cup A^m \cup A$, such that*

$$T \models \mathrm{dep}(x_1,\ldots,x_n,y) \vee \mathrm{dep}(u_1,\ldots,u_m,v) \quad \textit{iff} \quad T' \models \mathrm{dep}(x',y) \vee \mathrm{dep}(u',v).$$

The next lemma establishes the coherence of one of the simplest higher-arity disjunctions.

▶ **Lemma 29.** *The formula $\mathrm{dep}(x,z) \vee \mathrm{dep}(x,y,u)$ has coherence-level 4.*

Proof. To show that $\mathrm{dep}(x,z) \vee \mathrm{dep}(x,y,u)$ is not 3-coherent, notice that this formula is equivalent to $\mathrm{dep}(x,z) \vee \mathrm{dep}(x,u)$ on every team T that is constant on y, i.e., $s(y)=t(y)$ holds, for all $s,t \in T$. Thus, the supplement team T_f^y given by T in Figure 5 (a) (and with suitable variable renamings) and constant function $f(s)=a$ is a counterexample to the 3-coherence of $\mathrm{dep}(x,z) \vee \mathrm{dep}(x,y,u)$.

To show that $\mathrm{dep}(x,z) \vee \mathrm{dep}(x,y,u)$ is 4-coherent, consider an arbitrary team T with domain $\{x,z,y,u\}$. Without loss of generality, we may assume that T is constant on x. This is so because T can be split into sub-teams $T_{a_i}=\{s \in T \mid s(x)=a_i \in A\}$ that are constant on x. If T falsifies $\mathrm{dep}(x,z) \vee \mathrm{dep}(x,y,u)$, then at least one of these sub-teams also falsifies $\mathrm{dep}(x,z) \vee \mathrm{dep}(x,y,u)$. Indeed, if every sub-team T_{a_i} satisfied $\mathrm{dep}(x,z) \vee \mathrm{dep}(x,y,u)$, then their union T would also satisfy $\mathrm{dep}(x,z) \vee \mathrm{dep}(x,y,u)$, since every team T' consisting of two assignments $s \in T_{a_j}$ and $t \in T_{a_k}$ with $a_j \neq a_k$ satisfies both dependence atoms (because $s(x)=a_j \neq a_k = t(x)$).

Now, since T is constant on x, whether or not $T \models \mathrm{dep}(x,z)$ depends only on the values of z and so $\mathrm{dep}(x,z)$ acts like the constancy atom $\mathrm{dep}(z)$. Similarly, whether or not $T \models \mathrm{dep}(x,y,u)$, depends only on the values of y and u, and so $\mathrm{dep}(x,y,u)$ acts like the unary dependence atom into $\mathrm{dep}(y,u)$. Therefore, we have that $T \models \mathrm{dep}(x,z) \vee \mathrm{dep}(x,y,u)$ if and only if $T \models \mathrm{dep}(z) \vee \mathrm{dep}(y,u)$. As the latter disjunction was shown to be 4-coherent in Theorem 24, we have that $\mathrm{dep}(x,z) \vee \mathrm{dep}(x,y,u)$ is 4-coherent as well. ◀

We are now ready to state and prove a classification theorem for the complexity of the model-checking problem for disjunctions of dependence atoms of higher arities. Note that, when a dependence atom $\mathrm{dep}(y_1,\ldots,y_n,u)$ of higher arity is considered, then, by the semantics of dependence logic $\mathcal{D}$, the order $y_1,\ldots,y_n$ of the variables is immaterial.

▶ **Theorem 30.** *Let $\mathrm{dep}(x_1,\ldots,x_m,z)$ and $\mathrm{dep}(y_1,\ldots,y_n,u)$ be two dependence atoms such that $n \geq m$ and $n > 1$ (i.e., at least one of them is non-unary), and some of the variables of the first atom may coincide with some of the variables of the second. Then the following hold:*

(1.) *If neither the set $\{x_1,\ldots,x_m\}$ is contained in the set $\{y_1,\ldots,y_n\}$ nor the set $\{y_1,\ldots,y_n\}$ is contained in the set $\{x_1,\ldots,x_m\}$, then the problem*

$$\mathrm{MC}(\mathrm{dep}(x_1,\ldots,x_m,z) \vee \mathrm{dep}(y_1,\ldots,y_n,u))$$

is NL*-complete.*

(2.) *Otherwise (i.e., if* $\{x_1, \ldots, x_m\} \subseteq \{y_1, \ldots, y_n\}$*), the problem*

$$\mathrm{MC}(\mathrm{dep}(x_1, \ldots, x_m, z) \vee \mathrm{dep}(y_1, \ldots, y_n, u))$$

is in FO*, hence also in uniform* AC^0*.*

Proof. For the first part, membership in NL follows from Proposition 10. We show NL-hardness for the special case $z = u$, by exhibiting a logarithmic-space reduction from the problem $\mathrm{MC}(\mathrm{dep}(x, z) \vee \mathrm{dep}(y, z))$ (see Theorem 13). Let $\hat{x}$ be a variable in $\{x_1, \ldots, x_m\}$ but not in $\{y_1, \ldots, y_n\}$; similarly, let $\hat{y}$ be a variable in $\{y_1, \ldots, y_n\}$ but not in $\{x_1, \ldots, x_m\}$. Given a team T, we construct in logarithmic space a team T' as follows: for every $s \in T$, we put in T' an assignment s', such that $s'(z) = s(z)$, $s'(x_i) = s(x_i)$ if $x_i = \hat{x}$ and a otherwise, likewise $s'(y_i) = s(y_i)$ if $y_i = \hat{y}$ and a otherwise, where a is some fixed element in the range of T. Then the following equivalences hold:

$$\begin{aligned}
&T \models \mathrm{dep}(x, z) \vee \mathrm{dep}(y, z)\\
&\text{iff}\quad \exists T_1, T_2 \colon T = T_1 \cup T_2, T_1 \models \mathrm{dep}(x, z) \text{ and } T_2 \models \mathrm{dep}(y, z)\\
&\text{iff}\quad \exists T_1, T_2 \colon T = T_1 \cup T_2, \forall s_1, t_1 \in T_1 \colon s_1(x) = t_1(x) \Rightarrow s_1(z) = t_1(z) \text{ and}\\
&\quad \forall s_2, t_2 \in T_2 \colon s_2(y) = t_2(y) \Rightarrow s_2(z) = t_2(z)\\
&\text{iff}\quad \exists T_1', T_2' \colon T' = T_1' \cup T_2', \forall s_1', t_1' \in T_1' \colon s_1'(\hat{x}) = t_1'(\hat{x}) \Rightarrow s_1'(z) = t_1'(z) \text{ and}\\
&\quad \forall s_2', t_2' \in T_2' \colon s_2'(\hat{y}) = t_2'(\hat{y}) \Rightarrow s_2'(z) = t_2'(z)\\
&\text{iff}\quad \exists T_1', T_2' \colon T' = T_1' \cup T_2', T_1' \models \mathrm{dep}(x_1, \ldots, x_m, z) \text{ and } T_2' \models \mathrm{dep}(y_1, \ldots, y_n, z) \qquad (\star)\\
&\text{iff}\quad T' \models \mathrm{dep}(x_1, \ldots, x_m, z) \vee \mathrm{dep}(y_1, \ldots, y_n, z).
\end{aligned}$$

Equivalence $(\star)$ holds, because if $x_i \neq \hat{x}$, then $s'(x_i) = t'(x_i) = a$ (and similarly for the y_i's).

For the second part, we distinguish two cases, namely whether $m = n$ or $m < n$.

Case "$m = n$": The formula $\mathrm{dep}(x_1, \ldots, x_m, z) \vee \mathrm{dep}(x_1, \ldots, x_m, u)$ has coherence-level 4, because, via Corollary 28, the model-checking problem of $\mathrm{dep}(x_1, \ldots, x_m, z) \vee \mathrm{dep}(x_1, \ldots, x_m, u)$ is equivalent to that of $\mathrm{dep}(x, y) \vee \mathrm{dep}(x, z)$. Thus, by Proposition 9 and Theorem 22, we have that $\mathrm{MC}(\mathrm{dep}(x_1, \ldots, x_m, z) \vee \mathrm{dep}(y_1, \ldots, y_n, u))$ is in FO.

Case "$m < n$": Assume $y_{m+1}, \ldots, y_n$ are the variables that do not occur in the first dependence atom $\mathrm{dep}(x_1, \ldots, x_m, z)$. Let $x = (x_1, \ldots, x_m)$ and $y = (y_{m+1}, \ldots, y_n)$. Then, by Corollary 28, the disjunction $\mathrm{dep}(x_1, \ldots, x_m, z) \vee \mathrm{dep}(x_1, \ldots, x_m, y_{m+1}, \ldots, y_n, u)$ is equivalent to the disjunction $\mathrm{dep}(x, z) \vee \mathrm{dep}(x, y, u)$, which is 4-coherent by Lemma 29. Thus by Proposition 9, we have that $\mathrm{MC}(\mathrm{dep}(x_1, \ldots, x_m, z) \vee \mathrm{dep}(y_1, \ldots, y_n, u))$ is in FO. ◀

Observe that, whenever $\mathrm{MC}(\varphi)$ was shown to be in FO, the proof of Theorem 30 actually showed that φ is coherent. Combined with the earlier results about unary dependence atoms, we obtain the following characterization of coherence of disjunctions of dependence atoms.

▶ **Corollary 31.** *Let* φ *be a disjunction of two dependence atoms of arbitrary arities. Then* φ *is coherent if and only if* $\mathrm{MC}(\varphi)$ *is first-order definable.*

6 Outlook

In this paper, we carried out a systematic investigation of the model-checking problem for disjunctions of two dependence atoms. The work reported here suggests several different directions for future research, including the following:

1. Is it possible to classify the computational complexity of the model-checking problem for all quantifier-free formulas of dependence logic $\mathcal{D}$? In particular, is there a dichotomy theorem between NP-completeness and polynomial-time solvability for this problem? Note that it is conceivable that every problem in NP is polynomial-time equivalent to the model-checking problem of a quantifier-free $\mathcal{D}$-formula. In that case, such a dichotomy theorem would be impossible (unless P = NP), since by Ladner's theorem [12], if P $\neq$ NP, then there are problems in NP that are neither NP-complete nor are in P.
2. By Proposition 9, if φ is a coherent, quantifier-free $\mathcal{D}$-formula, then φ is equivalent to an FO-formula $\varphi^*(T)$ that includes a relation symbol T for the team. We conjecture that the converse is true, which would imply that coherence coincides with first-order definability for quantifier-free $\mathcal{D}$-formulas. Our results confirm this conjecture for the case in which φ is a disjunction of two dependence atoms.
3. Study the *implication problem* for disjunctions of two dependence atoms, i.e., given a finite set Σ of disjunctions of two dependence atoms and a disjunction ψ of two dependence atoms, does Σ logically imply ψ? For functional dependencies (i.e., single dependence atoms), the implication problem is solvable in polynomial time; moreover, there is a set of simple axioms, known as Armstrong's axioms [1, p. 186], for reasoning about functional dependencies. As argued in the introduction, disjunctions of two dependence atoms form a natural class of database dependencies that have not been studied in their own right thus far. Investigating the implication problem for disjunctions of two dependence atoms will enhance the interaction between dependence logic and database theory.

References

1. Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases.* Addison-Wesley, 1995. URL: `http://webdam.inria.fr/Alice/`.
2. Stephen A. Cook and Pierre McKenzie. Problems complete for deterministic logarithmic space. *J. Algorithms*, 8(3):385–394, 1987. `doi:10.1016/0196-6774(87)90018-6`.
3. Elias Dahlhaus. Reduction to NP-complete problems by interpretations. In *Logic and Machines*, volume 171 of *Lecture Notes in Computer Science*, pages 357–365. Springer, 1983. `doi:10.1007/3-540-13331-3_51`.
4. Ronald Fagin. Generalized first-order spectra and polynomial-time recognizable sets. In Richard Karp, editor, *Complexity of Computation*, number 7 in SIAM-AMS Proceedings, pages 43–73. SIAM-AMS, 1974.
5. Ronald Fagin and Moshe Y. Vardi. The Theory of Data Dependencies - A Survey. *Symposia in Applied Mathematics*, pages 19–71, 1986. `doi:10.1090/psapm/034/846853`.
6. Nicolas Fröhlich, Phokion G. Kolaitis, and Arne Meier. Disjunctions of two dependence atoms, 2025. `arXiv:2508.16146`.
7. Neil Immerman. *Descriptive complexity.* Graduate texts in computer science. Springer, 1999. `doi:10.1007/978-1-4612-0539-5`.
8. Jarmo Kontinen. Coherence and computational complexity of quantifier-free dependence logic formulas. *Studia Logica*, 101(2):267–291, 2013. `doi:10.1007/s11225-013-9481-8`.
9. Juha Kontinen, Sebastian Link, and Jouko A. Väänänen. Independence in database relations. In *WoLLIC*, volume 8071 of *Lecture Notes in Computer Science*, pages 179–193. Springer, 2013. `doi:10.1007/978-3-642-39992-3_17`.
10. Juha Kontinen and Jouko A. Väänänen. On definability in dependence logic. *Journal of Logic, Language and Information*, 18(3):317–332, 2009. `doi:10.1007/s10849-009-9082-0`.
11. Juha Kontinen and Jouko A. Väänänen. Axiomatizing first-order consequences in dependence logic. *Ann. Pure Appl. Logic*, 164(11):1101–1117, 2013. `doi:10.1016/j.apal.2013.05.006`.
12. Richard E. Ladner. On the structure of polynomial time reducibility. *J. ACM*, 22(1):155–171, 1975. `doi:10.1145/321864.321877`.

13 Omer Reingold. Undirected connectivity in log-space. *J. ACM*, 55(4):17:1–17:24, 2008. `doi:10.1145/1391289.1391291`.

14 Jouko A. Väänänen. *Dependence Logic – A New Approach to Independence Friendly Logic*, volume 70 of *London Mathematical Society student texts*. Cambridge University Press, 2007. URL: `http://www.cambridge.org/de/knowledge/isbn/item1164246/?site_locale=de_DE`.

15 Moshe Y. Vardi. The complexity of relational query languages (extended abstract). In *STOC*, pages 137–146. ACM, 1982. `doi:10.1145/800070.802186`.

A Appendix

▶ **Theorem 13** ($\star$)**.** *The following statements are true:*
(1.) *The problem* $\mathrm{MC}(\mathrm{dep}(x,z) \vee \mathrm{dep}(y,z))$ *is* NL*-complete.*
(2.) *The problem* $\mathrm{MC}(\mathrm{dep}(x,y) \vee \mathrm{dep}(y,z))$ *is* NL*-complete.*

Proof. Membership follows directly from Proposition 10. We now give the proof of NL-hardness for **(1.)**. The proof works analogous for **(2.)**.

We show 2SAT $\leq_m^{\log}$ $\mathrm{MC}(\mathrm{dep}(x,z) \vee \mathrm{dep}(y,z))$. Suppose $\theta(p_1,\ldots,p_m)$ is an instance of 2SAT of the form $\bigwedge_{i\in I} E_i$, with conjuncts $E_i = (\ell_{i_1} \vee \ell_{i_2})$, where ℓ_{i_1}, ℓ_{i_2} are different literals, i. e., $\ell_{i_1} \neq \ell_{i_2}$. This can easily be archived by replacing conjuncts of the form $(\ell \vee \ell)$ with $(\ell \vee q)$ and $(\ell \vee \neg q)$, where $q \notin \{p_1,\ldots,p_m\}$ is a new proposition.

We will construct a team T, such that the following are equivalent:

- $T \models \mathrm{dep}(x,z) \vee \mathrm{dep}(y,z)$.
- $\theta(p_1,\ldots,p_m)$ is satisfiable.

For each conjunct E_i, we create a team T_{E_i} with two assignments s_{i_1} and s_{i_2} of domain $\{x,y,z\}$, that encode the literals of E_i. To this end, variable x ranges over the propositions $\{p_1,\ldots,p_m\}$, variable y ranges over clause (index) I, and variable z ranges over $\{\, 0_k, 1_k \mid 1 \leq k \leq m \,\}$, such that each proposition has its own two truth values. This is necessary to ensure that z has two different values in each T_{E_i}. Thus, the dependence atom $\mathrm{dep}(y,z)$ makes sure we have to choose at least one of the assignments from each T_{E_i} into the subset of T, that will satisfy $\mathrm{dep}(x,z)$, thereby encoding the assignment that satisfies θ.

Each literal ℓ_{i_j} in E_i, gives rise to one assignment. For example, the team T_{E_i} for a clause $(p_k \vee p_j)$ is the one in the middle in Figure 1.

Now T is the union $\bigcup_{i\in I} T_{E_i}$. Suppose $\theta(p_1,\ldots,p_m)$ is satisfiable. Then there exists an assignment $F\colon \{p_1,\ldots,p_m\} \to \{0,1\}$, that satisfies $\theta(p_1,\ldots,p_m)$. We write $F(p_k)_k$ to denote the truth values $\{0_k, 1_k\}$ used by variable z. For example, if $F(p_k) = 0$, then $F(p_k)_k = 0_k$. Define the partition of the team T into two sets in the following way:

$$T_1 = \{s \in T \mid s(x) = p_k \text{ and } s(z) = F(p_k)_k\}, \qquad T_2 = T \setminus T_1$$

The assignments in T that agree with the assignment F are chosen to T_1. Since F evaluates $\bigwedge_{i\in I} E_i$ true, it satisfies every conjunct E_i. Thus T_1 contains at least one assignment from each T_{E_i}. Thus there will be at most one tuple from each T_{E_i} left to T_2. This T_2 trivially satisfies $\mathrm{dep}(y,z)$ since all tuples in T_2 disagree on y, i. e., for all $s,t \in T_2$ it is true that $s(y) \neq t(y)$, if $s \neq t$.

Next we will show that T_1 satisfies $\mathrm{dep}(x,z)$. Let $s,t \in T_1$, such that $s(x) = t(x) = p_k$. Then by the definition of T_1 it holds that $s(y) = F(p_k)_k = t(y)$ holds. Thus $T_1 \models \mathrm{dep}(x,z)$.

The other direction: Suppose $T \models \mathrm{dep}(x,z) \vee \mathrm{dep}(y,z)$. Then there is a partition of T into T_1 and T_2, such that $T_1 \models \mathrm{dep}(x,z)$ and $T_2 \models \mathrm{dep}(y,z)$. We will define the assignment $F\colon \{p_1,\ldots,p_m\} \to \{0,1\}$ in the following way:

- If there exists $s \in T_1$, such that $s(x) = p_k$, then $F(p_k)_k = s(z)$.
- Otherwise, if for all $s \in T_1$ it holds $s(x) \neq p_k$, then $F(p_k) = 1$.

Let us show that F is a function, which satisfies $\theta(p_1, \dots, p_m)$:

1. Clearly, $\mathrm{Dom}(F) = \{p_1, \dots, p_m\}$ and $\mathrm{Rng}(F) = \{0, 1\}$.
2. F is a function: Let $p_k \in \{p_1, \dots, p_m\}$. Suppose there exists $s, t \in T_1$, such that $s(x) = t(x) = p_k$ holds. Since $T_1 \models \mathrm{dep}(x, z)$ holds, it follows that $s(z) = t(z)$ holds. Suppose there are no $s \in T_1$, such that $s(x) = p_k$. Then by definition of F it holds that $F(p_k) = 1$.
3. F satisfies $\theta(p_1, \dots, p_m)$: Note that y is constant and z is assigned different values by each tuple in each T_{E_i}. Thus T_1 contains at least one of the tuples from each T_{E_i}. Let $s \in T_{E_i}$, such that $s \in T_1$. Recall, that in every assignment the values $s(z)$ encodes the truth value of $s(x)$ such that E_i is satisfied. Since s agrees with F, it holds that $F(\ell_{i_j})_{i_j} = s(z)$, which implies that $F(E_i) = 1$.

Each conjunct E_i, of θ gives rise to a constant size team of two assignments with domain $\{x, y, z\}$. Thus the team T can be constructed in L for each θ. ◀

▶ **Lemma 15 (⋆).** *Let T be a team with domain $\{x, y\}$ such that $\mathrm{Rng}_x(T) \cap \mathrm{Rng}_y(T) = \emptyset$. The following statements are equivalent:*

(1.) *$T \models \mathrm{dep}(x, y) \vee \mathrm{dep}(y, x)$.*

(2.) *G_T can be transformed into a directed graph $G'_T = (V_T, E'_T)$ such that*

- **(a)** *each node has at most one out-going edge, and*
- **(b)** *for each $\{u, v\} \in E_T$ either $(u, v) \in E'_T$ or $(v, u) \in E'_T$.*

(3.) *For each connected component $C = (V_C, E_C)$ of G_T, we have that $|E_C| \leq |V_C|$.*

Proof. We will prove the equivalences as follows.

(1.) ⇒ (2.) Assume, w.l.o.g., $T = T_1 \uplus T_2$, such that $T_1 \models \mathrm{dep}(x, y)$ and $T_2 \models \mathrm{dep}(y, x)$. Then the directed graph $G'_T = (V_T, E'_T)$ with $E'_T = \{\, (s(x), s(y)) \mid s \in T_1 \,\} \cup \{\, (s(y), s(x)) \mid s \in T_2 \,\}$ has properties (a) and (b).

For (a), wrongly assume, w.l.o.g., that there are two edges $e_1 = (a, b)$ and $e_2 = (a, b')$ with $b \neq b'$ and a, b, b' values in $\mathrm{Rng}(T)$ (the same argument could be made for the second component being a and the first components being different). Now, these edges correspond to assignments $s(x) = a$, $s(y) = b$, and $s'(x) = a$, $s'(x) = b'$. Both $s, s' \in T_1$ and would violate $\mathrm{dep}(x, y)$ which is a contradiction.

For (b), notice that each $\{u, v\}$ corresponds to a single assignment $s \in T$. By assumption, s is either in T_1 or T_2, whence it has to appear in E'_T in either direction.

(2.) ⇒ (1.) Let $G'_T = (V_T, E'_T)$ be a directed graph, where each node has at most one out-going edge. Define $T_1 = \{s \in T \mid (s(x), s(y)) \in E'_T\}$ and $T_2 = \{s \in T \mid (s(y), s(x)) \in E'_T\}$. Then, $T_1 \models \mathrm{dep}(x, y)$, because each $s(x)$ appears only once by assumption, so the dependency is not violated. By analogous arguments we have that $T_2 \models \mathrm{dep}(y, x)$.

(2.) ⇒ (3.) We make a case distinction. If each node in G'_T has exactly one out-going edge, then $|V_T| = |E'_T| \overset{(b)}{=} |E_T|$ and likewise $|E_C| = |V_C|$ for each connected component C of G_T. Otherwise, if some nodes have no out-going edge, then $|E_C| < |V_C|$.

(3.) ⇒ (2.) A connected component in G_T has at least $|V_C| - 1$ edges (otherwise it would not be connected). So there are only two cases:

- **(i)** If $|E_C| = |V_C| - 1$, then C is a (spanning-)tree. Define G'_T by orienting the edges from the leaves to the root, i.e., bottom-up. Since each node has one parent, each node has one outgoing edge (except for the root which has no outgoing edge).
- **(ii)** If $|E_C| = |V_C|$, then C contains *exactly one* cycle $\mathcal{C}$, because C is connected and has one more edge than its spanning-tree. Choose any orientation of $\mathcal{C}$ (clockwise or counter-clockwise). Afterwards, all nodes in $C \setminus \mathcal{C}$ (hence outside of the cycle $\mathcal{C}$) form tree-like connected components and we can proceed as in **(i)**: There is one node directly connecting to $\mathcal{C}$ which will become the root, and its outgoing edge connects to a node in the cycle $\mathcal{C}$. ◀

▶ **Lemma 17** ($\star$). *Let T be a team with domain $\{x, y\}$. If the size $|T| > |\operatorname{Rng}_x(T)| + |\operatorname{Rng}_y(T)|$, then $T \not\models \operatorname{dep}(x, y) \vee \operatorname{dep}(y, x)$.*

Proof. Assume $T \models \operatorname{dep}(x, y) \vee \operatorname{dep}(y, x)$ and $|T| > |\operatorname{Rng}_x(T)| + |\operatorname{Rng}_y(T)|$, then there exists a split $T = T_1 \cup T_2$ such that $T_1 \models \operatorname{dep}(x, y)$ and $T_2 \models \operatorname{dep}(y, x)$. Clearly, $|T_1| \leq |\operatorname{Rng}_y(T)|$ and $|T_2| \leq |\operatorname{Rng}_y(T)|$, because otherwise each of T_1 and T_2 would not satisfy the respective dependence atom. That is true, because then there would be a pair $s, t \in T_1$ with $s(x) = t(x)$ and $s(y) \neq t(y)$, respectively $s, t \in T_2$ with $s(y) = t(y)$ and $s(x) \neq t(x)$. Furthermore, we have that $|T| \leq |T_1| + |T_2|$. As a result, we get $|T| \leq |T_1| + |T_2| \leq |\operatorname{Rng}_x(T)| + |\operatorname{Rng}_y(T)|$, which is a contradiction to $|T| > |\operatorname{Rng}_x(T)| + |\operatorname{Rng}_y(T)|$. ◀

▶ **Definition 19.** *Let ϕ be a 2CNF formula with variables $\operatorname{Var}(\phi) = \{x_1, \ldots, x_n\}$, such that:*

(1.) *Every clause is monotone, i.e., it is of the form $(x_i \vee x_j)$ or of the form $(\bar{x}_i \vee \bar{x}_j)$. The first type is called* positive*, the second type* negative*.*

(2.) *No two "dual" clauses $(x_i \vee y_j)$, $(\bar{x}_i \vee \bar{x}_j)$ appear in ϕ.*

(3.) *Transitivity holds: if $(x_i \vee x_j)$ and $(x_j \vee x_k)$ are clauses of ϕ, then so is $(x_i \vee x_k) \in \phi$; furthermore, if $(\bar{x}_i \vee \bar{x}_j)$ and $(\bar{x}_j \vee \bar{x}_k)$ are clauses of ϕ, then so is $(\bar{x}_i \vee \bar{x}_k) \in \phi$.*

We call such 2CNF formulas monotone transitive dual-free *and abbreviate the corresponding satisfiability problem by* mtdf-2SAT*.*

We call a set $\{v_1, \ldots, v_k\}$ of variables in ϕ a *(variable-)clique*, if for each pair $v_i, v_j, i \neq j$ it is true that either $(v_i \vee v_j)$ or $(\neg v_i \vee \neg v_j)$ is a clause in ϕ. Note that, by the transitivity (3.), the variables of a mtdf-2SAT instance appear in at most two cliques, once positive and once negative. Thus we can represent ϕ, by listing its positive and negative cliques.

▶ **Lemma 20** ($\star$). *The problems* $\operatorname{MC}(\operatorname{dep}(x, y) \vee \operatorname{dep}(y, x))$ *and* mtdf-2SAT *are equivalent via first-order reductions.*

Proof. We present reductions between $\operatorname{MC}(\operatorname{dep}(x, y) \vee \operatorname{dep}(y, x))$ and mtdf-2SAT (for illustrations, see Figure 7). Let $\tau_t = (T^3)$ be the vocabulary of structures where T encodes a team such that $T(i, j, k)$ is true if and only if $s_i(x_j) = a_k$, where i, j, k are natural numbers. Note that this differs from the encoding $\operatorname{Rel}(T)$, used in the proof of Theorem 18, in that it explicitly contains the index of assignments. This additional information is necessary for this reduction. Then $\operatorname{MC}(\operatorname{dep}(x, y) \vee \operatorname{dep}(y, x)) \colon \operatorname{STRUC}[\tau_t] \to \{0, 1\}$ is the Boolean query that is true if and only if T is a valid team and $T \models \operatorname{dep}(x, y) \vee \operatorname{dep}(y, x)$.

Further let $\tau_{pn} = (P^2, N^2)$ be the vocabulary of structures for mtdf-2SAT, where $P(v, c)$ encodes positive variables that occur in the same clique c and $N(v, c)$ encodes negative variables. Then mtdf-2SAT: $\operatorname{STRUC}[\tau_{pn}] \to \{0, 1\}$ is the Boolean query that is true if and only if P and N encode a mtdf-2CNF formula ϕ and there is some assignment $\mathfrak{I}$ such that $\mathfrak{I} \models \phi$.

The first-order reduction $I_{\text{tpn}} : \operatorname{STRUC}[\tau_t] \to \operatorname{STRUC}[\tau_{pn}]$ is as follows:

$$I_{\text{tpn}} \coloneqq \lambda_{vc}\langle \texttt{true}, \psi_P, \psi_N \rangle, \qquad \psi_P(v, c) \coloneqq T(v, 0, c), \qquad \psi_N(v, c) \coloneqq T(v, 1, c),$$

where `true` means the universe is the same. We show that $A \in \operatorname{MC}(\operatorname{dep}(x, y) \vee \operatorname{dep}(y, x)) \Leftrightarrow I_{\text{tpn}}(A) \in$ mtdf-2SAT. Assume A encodes a team T such that $T = T_1 \cup T_2$, where $T_1 \models \operatorname{dep}(x, y)$ and $T_1 \models \operatorname{dep}(y, x)$. Then the assignment $\mathfrak{I}$ with $\mathfrak{I}(x_i) = 1$, if $s_i \in T_1$ and $\mathfrak{I}(x_i) = 0$, if $s_i \in T_2$ satisfies the formula encodes by $I_{\text{tpn}}(A)$. This is because, in a positive clique, at most one variable is false. Thus, in all clauses, at least one variable is true. The opposite holds for negative cliques.

T	x	y
s_0	0	0
s_1	0	1
s_2	0	2
s_3	1	1
s_4	2	1
s_5	2	2

$$\leftrightsquigarrow \quad \begin{array}{l} T(0,0,0),\ T(0,1,0),\\ T(1,0,0),\ T(1,1,1),\\ T(2,0,0),\ T(2,1,2),\\ T(3,0,1),\ T(3,1,1),\\ T(4,0,2),\ T(4,1,1),\\ T(5,0,2),\ T(5,1,2) \end{array} \quad \leftrightsquigarrow \quad \begin{array}{l} P(0,0),\ N(0,0),\\ P(1,0),\ N(1,1),\\ P(2,0),\ N(2,2),\\ P(3,1),\ N(3,1),\\ P(4,2),\ N(4,1),\\ P(5,2),\ N(5,2) \end{array} \quad \leftrightsquigarrow \quad \begin{array}{c} (x_0 \vee x_1),\ (x_1 \vee x_0),\\ (x_1 \vee x_2),\ (x_2 \vee x_1),\\ (x_0 \vee x_2),\ (x_2 \vee x_0),\\ (x_4 \vee x_5),\ (x_5 \vee x_4),\\ (\neg x_1 \vee \neg x_3),\ (\neg x_3 \vee \neg x_1),\\ (\neg x_3 \vee \neg x_4),\ (\neg x_4 \vee \neg x_3),\\ (\neg x_1 \vee \neg x_4),\ (\neg x_4 \vee \neg x_1),\\ (\neg x_2 \vee \neg x_5),\ (\neg x_5 \vee \neg x_2) \end{array}$$

Figure 7 Example reduction for Lemma 20. Team T gets encodes via structure $\langle\{0,1,2,3,4,5\}, T^3\rangle$ that gets mapped to the structure $\langle\{0,1,2,3,4,5\}, P^2, N^2\rangle$ which encodes a mtdf-2CNF formula ϕ; and vice versa. Notice how the split $\{s_1, s_3, s_5\} \models \mathrm{dep}(x,y)$ and $\{s_0, s_2, s_4\} \models \mathrm{dep}(y,x)$ corresponds to the assignment $\mathfrak{I}(x_1) = \mathfrak{I}(x_3) = \mathfrak{I}(x_5) = 1$ and $\mathfrak{I}(x_0) = \mathfrak{I}(x_2) = \mathfrak{I}(x_4) = 0$.

Now for the reduction in the other direction. Consider the following first-order reduction $I_{\mathrm{pnt}}\colon \mathrm{STRUC}[\tau_{pn}] \to \mathrm{STRUC}[\tau_t]$ given by:

$$I_{\mathrm{pnt}} \coloneqq \lambda_{sxa}\langle \mathtt{true}, \psi_T\rangle, \qquad \psi_T(s,x,a) \coloneqq (x = 0 \to N(s,a)) \wedge (x = 1 \to P(s,a)).$$

Notice that I_{pnt} is simply the inverse of I_{tpn}, i. e., $I_{\mathrm{pnt}} = I_{\mathrm{tpn}}^{-1}$. ◀

▶ **Theorem 24** ($\star$). *Each of the following formulas has coherence-level* 4*:*
(1.) $\mathrm{dep}(x) \vee \mathrm{dep}(y,z)$*;* **(2.)** $\mathrm{dep}(x) \vee \mathrm{dep}(x,y)$*;* **(3.)** $\mathrm{dep}(y) \vee \mathrm{dep}(x,y)$.

Proof. We first demonstrate that $\mathrm{dep}(x) \vee \mathrm{dep}(y,z)$, $\mathrm{dep}(x) \vee \mathrm{dep}(x,y)$ and $\mathrm{dep}(y) \vee \mathrm{dep}(x,y)$ are not 3-coherent.

(1.) Consider the team T in Figure 5 (b). We have that $T \not\models \mathrm{dep}(x) \vee \mathrm{dep}(y,z)$ which is straightforward to see; consider for example the splits $\{s_1, s_2\} \models \mathrm{dep}(x)$ or $\{s_3, s_4\} \models \mathrm{dep}(x)$ and notice that in both cases $\mathrm{dep}(y,z)$ is falsified by $\{s_3, s_4\}$ and $\{s_1, s_2\}$ respectively.

Now, all 3-element sub-teams have a split $T' = T_1 \cup T_2$, where T_1 is of size two and $T_1 \models \mathrm{dep}(x)$ and T_2 is of size one and therefore satisfies any dependence atom.

(2.) Not 3-coherent, because of the team in Figure 5 (c).

(3.) Not 3-coherent, because of the team in Figure 5 (d).

For coherence it suffices to only consider $\mathrm{dep}(x) \vee \mathrm{dep}(y,z)$. The other two cases directly follow from the first.

We show that if $T \not\models \mathrm{dep}(x) \vee \mathrm{dep}(y,z)$, then there must be a sub-team $T' \subseteq T$ with $|T'| \leq 4$ and $T' \not\models \mathrm{dep}(x) \vee \mathrm{dep}(y,z)$. Start with a pair of assignments $s_1, s_2 \in T$ for which $\{s_1, s_2\} \not\models \mathrm{dep}(y,z)$ holds, i. e., $s_1(y) = s_2(y)$ and $s_2(z) \neq s_2(z)$. Such a pair has to exist in T to falsify the disjunction. Now, there are two cases for the assignment of x.

Case 1: The assignments s_1 and s_2 have the same value for x, i. e., $s_1(x) = s_2(x)$. Then for T to falsify the disjunction, there must be a second pair $\{s_3, s_4\} \not\models \mathrm{dep}(y,z)$ with $s_3(x) \neq s_1(x)$ and $s_4(x) \neq s_1(x)$. but then $T' = \{s_1, s_2, s_3, s_4\}$ and $T' \not\models \mathrm{dep}(x) \vee \mathrm{dep}(y,z)$. To see this, assume there is a split $T_1 \cup T_2 = \{s_1, s_2, s_3, s_4\}$ with $T_1 \models \mathrm{dep}(x)$ and $T_2 \models \mathrm{dep}(y,z)$. If $s_1 \in T_1$, then $s_3 \in T_2$, but s_4 cannot be in T_1 or T_2. If $s_1 \in T_2$, then $s_2 \in T_1$, therefore s_3 and s_4 must be in T_2 which is not possible. Therefore no split exists.

Case 2: The assignments s_1 and s_2 have different values for x, i. .e, $s_1(x) \neq s_2(x)$. Then the pair satisfies neither $\mathrm{dep}(x)$ nor $\mathrm{dep}(y,z)$, so in any potential split $s_1 \in T_1$ implies $s_2 \in T_2$ and vice versa. Since the original team falsified the disjunction, there must be an assignment $s_3 \in T$ that can neither be in T_1 nor T_2 if $s_1 \in T_1$ and $s_2 \in T_2$. In the same way there must be an assignment $s_4 \in T$ that is not in T_1 or T_2 for $s_1 \in T_2$ and $s_2 \in T_1$. Therefore $T' = \{s_1, s_2, s_3, s_4\}$ also has no split satisfying $\mathrm{dep}(x) \vee \mathrm{dep}(y,z)$. ◀

T_n	x	y
s_0	1	$\frac{n}{2}$
s_1	1	1
s_2	1	2
s_3	2	2
s_4	2	3
$\vdots$	$\vdots$	$\vdots$
s_{n-1}	$\frac{n}{2}$	$\frac{n}{2}$
s_n	$\frac{n}{2}$	1

T'	x	y
s_0	1	$\frac{n}{2}$
s_3	2	2
$\vdots$	$\vdots$	$\vdots$
s_n	$\frac{n}{2}$	1
s_1	1	1
s_2	1	2
$\vdots$	$\vdots$	$\vdots$
s_{n-1}	$\frac{n}{2}$	$\frac{n}{2}$

T_n	x	y	z
s_0	1	$\frac{n}{2}$	3
s_1	1	1	1
s_2	1	2	2
s_3	2	2	1
s_4	2	3	2
$\vdots$	$\vdots$	$\vdots$	$\vdots$
s_{n-1}	$\frac{n}{2}$	$\frac{n}{2}$	1
s_n	$\frac{n}{2}$	1	2

T'	x	y	z
s_0	1	$\frac{n}{2}$	3
s_3	2	2	1
$\vdots$	$\vdots$	$\vdots$	$\vdots$
s_n	$\frac{n}{2}$	1	2
s_1	1	1	1
s_2	1	2	2
$\vdots$	$\vdots$	$\vdots$	$\vdots$
s_{n-1}	$\frac{n}{2}$	$\frac{n}{2}$	1

Figure 8 Team T_n in the proof of Theorem 25, and constructed split of an example sub-team $T' = T_n \setminus \{s_4\}$. On the left for $\text{dep}(x, y) \vee \text{dep}(y, x)$ and on the right for $\text{dep}(x, z) \vee \text{dep}(y, z)$.

▶ **Theorem 25** ($\star$). *The following three formulas are incoherent:*
(1.) $\text{dep}(x, y) \vee \text{dep}(y, x)$*;* **(2.)** $\text{dep}(x, y) \vee \text{dep}(y, z)$*;* **(3.)** $\text{dep}(x, z) \vee \text{dep}(y, z)$.

Proof. **(1.)** We show that for all even $n \in \mathbb{N}$, there is a team T of size $|T| = n + 1$ with $T \not\models \text{dep}(x, y) \vee \text{dep}(y, x)$, but for all n-element sub-teams $T' \models \text{dep}(x, y) \vee \text{dep}(y, x)$ holds.

Let the team T_n be as depicted in Figure 8. We show that it is impossible to split $T_n = T_1 \cup T_2$ such that $T_1 \models \text{dep}(x, y)$ and $T_2 \models \text{dep}(y, x)$. Start with assignment s_1 and choose $s_1 \in T_1$. Next, $s_2 \in T_2$ is the only choice, because $T_1 = \{s_1, s_2\} \not\models \text{dep}(x, y)$. Then $s_3 \in T_1$, because $T_2 = \{s_2, s_3\} \not\models \text{dep}(y, x)$, but $T_1 = \{s_1, s_3\} \models \text{dep}(x, y)$. Continue this procedure until all assignments but s_0 are either in T_1 or T_2. Now, s_0 cannot be in T_1, because $\{s_0, s_1\} \not\models \text{dep}(x, y)$ and it cannot be in T_2, because $\{s_0, s_{n-2}\} \not\models \text{dep}(y, x)$. If instead we chose $s_1 \in T_2$ and repeated the process backwards, that is continued with s_n, then $\{s_0, s_2\} \not\models \text{dep}(x, y)$ and $\{s_0, s_{n-1}\} \not\models \text{dep}(y, x)$. Since s_1 has to be in either T_1 or T_2, we can conclude that no split exists and $T_n \not\models \text{dep}(x, y) \vee \text{dep}(y, x)$.

Next, we show that all sub-teams of size n satisfy the disjunction. The sub-team $T_n \setminus \{s_0\} \models \text{dep}(x, y) \vee \text{dep}(y, x)$ given the split described above. Otherwise, let $T' = T_n \setminus \{s_i\}$ and consider the split above where $s_1 \in T_1$. We define a new split for T' as follows

$$T'_1 = \{s_j \in T_1 \mid 1 < j < i\} \cup \{s_j \in T_2 \mid j > i\} \cup \{s_0\}$$
$$T'_2 = \{s_j \in T_2 \mid 1 < j < i\} \cup \{s_j \in T_1 \mid j > i\} \cup \{s_1\}$$

Now, $T'_1 \models \text{dep}(x, y)$ and $T'_2 \models \text{dep}(y, x)$.

We will present the case that $s_i \in T_1$; the case $s_i \in T_2$ follows analogously. If $s_i \in T_1$, then s_{i+1} has to be in T_2 as shown above. Now, since $s_i \notin T'$, it is possible that $s_{i+1} \in T'_1$. From this $s_{i+2} \in T'_2, s_{i+3} \in T'_1, \ldots$ follows immediately. This chain leads to $s_n \in T'_1$, which allows $s_1 \in T'_2$, which in turn makes $s_0 \in T'_1$ possible. Therefore we have that $T'_1 \models \text{dep}(x, y)$ and $T'_2 \models \text{dep}(y, x)$.

We have shown that for every even $n \in \mathbb{N}$ we can construct a team such that $\text{dep}(x, y) \vee \text{dep}(y, x)$ is not n-coherent, that is $\text{dep}(x, y) \vee \text{dep}(y, x)$ is incoherent.

(2.) Extend the team T_n in **(1.)** with variable z, such that $s(z) = s(x)$ for all $s \in T_n$. Then incoherence of $\text{dep}(x, y) \vee \text{dep}(y, z)$ follows immediately from the incoherence of $\text{dep}(x, y) \vee \text{dep}(y, x)$.

(3.) For $\text{dep}(x, z) \vee \text{dep}(y, z)$ extend T_n with variable z such that if $s(x) = s'(x)$ or $s(y) = s'(y)$, then $s(z) \neq s'(z)$ for all $s, s' \in T$. For example see team T_n in Figure 8. It is easy to check, that the arguments of **(1.)** also hold true in such a team for $\text{dep}(x, z) \vee \text{dep}(y, z)$. ◀