Database Theory – ICDT 2012

15th International Conference
on Database Technology
Berlin, Germany, March 26–29, 2012
Proceedings

Editor:
Alin Deutsch (*University of California, San Diego, USA)*

Database Theory – ICDT 2012
Proceedings of the 15th International Conference
on Database Theory
Berlin, Germany, March 26–29, 2012

Editor:
Alin Deutsch

The Association for Computing Machinery
2 Penn Plaza, Suite 701
New York, NY, 10121-0701

ACM ISBN: 978-1-4503-0791-8

Table of Contents

Foreword

The papers in this volume were presented at the 15th International Conference on Database Theory (ICDT'12), held in Berlin, Germany, March 26-28, 2012. Starting in 2009, ICDT was held jointly with the EDBT (Extending Database Technology) conference. EDBT took place on March 27-29, 2012. The joint conference also included a series of affiliated workshops, held on March 30, 2012.

In response to the Call for Papers, 60 submissions were received by the submission deadline of July 29, 2011. All were submitted electronically through the EasyChair conference management tool. EasyChair was also used for the virtual Program Committee meeting, whose deliberations where held exclusively electronically. The Program Committee selected 22 papers for presentation. Among them, the Program Committee selected the paper “Learning Schema Mappings”, by Balder Ten Cate, Victor Dalmau and Phokion Kolaitis for the ICDT Best Paper Award, and the paper "Validating XML documents in the Streaming Model with External Memory", by Christian Konrad and Frederic Magniez, for the ICDT Best Newcomer Paper Award.

In addition, there were four ICDT/EDBT keynote speakers: Michael Carey (UC Irvine), Wenfei Fan (University of Edinburgh), Erich Graedel (RWTH Aachen University) and Alon Halevy (Google).

Editor

Alin Deutsch, *University of California, San Diego, USA*

Program Committee Members

Research

Reviewers

Serge Abiteboul	Marcelo Arenas	Michael Benedikt
Bogdan Cautis	Edith Cohen	Rada Chirkova
Claire David	Daniel Deutch	Alin Deutsch
Ronald Fagin	Floris Geerts	Gosta Grahne
Richard Hull	Daniel Kifer	Maurizio Lenzerini
Wim Martens	Maarten Marx	Tova Milo
Anca Muscholl	Dan Suciu	Jan Van den Bussche

External Reviewers

Yael Amsterdamer	Pablo Barceló	Leopoldo Bertossi
Abhishek Bhowmick	Johanna Björklund	Pierre Bourhis
Diego Calvanese	Anindya De	Alan Fakete
Gaelle Fontaine	Achille Fokoue	Olivier Gauwin
Shiva Kasiviswanathan	Benny Kimelfeld	Paraschos Koutris
Slawomir Lasota	Katrina Ligett	Johann Makowsky
Silviu Maniu	Alexandra Meliou	Filip Murlak
Frank Neven	Kobbi Nissim	Adrian Onet
Jorge Pérez	Juan Reutter	Riccardo Rosati
Cristina Sirangelo	Alex Thomo	David Toman
Yannis Velegrakis	Victor Vianu	Jef Wijsen
Yuqing Wu	Thomas Zeume	

Dependence, Independence, and Incomplete Information

Erich Grädel
Mathematische Grundlagen der Informatik,
RWTH Aachen University
graedel@logic.rwth-aachen.de

Jouko Väänänen
Department of Mathematics and Statistics,
University of Helsinki and
ILLC, University of Amsterdam
jouko.vaananen@helsinki.fi

ABSTRACT

Dependence logic, introduced by Väänänen, is the extension of first-order logic by atomic statements about (functional) dependencies of variables. An important feature of this logic is a model-theoretic semantics that, contrary to Tarski semantics, is not based on single assignments (mapping variables to elements of a structure) but on *sets* of assignments. Sets of assignments are called teams and the semantics is called team semantics.

By focussing on *independence* rather than depencence, we have proposed a new logic, called independence logic, based on atomic formulae $\overline{x} \perp_{\overline{z}} \overline{y}$ which intuitively say that the variables $\overline{x}$ are independent from the variables $\overline{y}$ whenever the variables $\overline{z}$ are kept constant. We show that $\overline{x} \perp_{\overline{z}} \overline{y}$ gives rise to a natural logic capable of formalizing basic intuitions about independence and dependence. We contrast this with dependence logic and show that independence logic has strictly more expressive power. Further, we will discuss game-theoretic semantics, expressive power, and complexity of dependence and independence logic.

1. INTRODUCTION

Statements about dependence or independence, such as *"x depends on y"* or *"the variables x and y are independent"* are of a substantially different nature than, say, statements about arithmetical properties auch as *"x divides y"*. To make sense of the latter we fix a structure $\mathfrak{A}$ in which the notion of divisibility has a well-defined meaning, and an assignment s mapping x and y to values in $\mathfrak{A}$, and we can then determine whether or not *"x divides y"* is true in $\mathfrak{A}$ for the assignment s, i.e., whether $\mathfrak{A} \models_s$ *"x divides y"*. Dependence and independence are concepts of a different kind. They do not manifest themselves in the presence of a single assignment but only for larger amount of data, given by a *set* of assignments or, equivalently, by a table or a relation in a database. Accordingly, model-theoretic semantics (also known as compositial semantics as opposed to game-theoretic semantics), for logics of dependence or independence refer to structures together with a set of assigments and thus differ substantially from the classical Tarski semantics of first-order logic, second-order logic and similar formalisms.

Logics of dependence and independence (often called logics of imperfect information) go back to the work of Henkin [10], Enderton [5], Walkoe [17], Blass and Gurevich [3], and others on partially order (or Henkin-) quantifiers, whose semantics can be naturally described in terms of games of imperfect information. A next step in this direction were the independence-friendly (IF) logics by Hintikka and Sandu [11] that incorporate explicit dependencies of quantifiers on each other. Again the semantics is usually given in game-theoretic terms. It had repeatedly be claimed that a compositional semantics, defined by induction on the construction of formulae, could not be given for IF-logic. However, this claim had never been made precise, let alone proved. In fact the claim was later refuted by Hodges [12] who presented a compositional semantics for IF-logic in terms of what he called *trumps*, which are sets of assignments to a fixed finite set of variables. The question of why logics of imperfect information need semantics based on sets of assignments is further discussed by Hodges in [13].

In 2007, Väänänen [16] proposed a new approach. Rather than stating dependencies or independencies as annotations of quantifiers, he proposed to consider dependence as an atomic formula, denoted $=\!(x_1, \dots, x_m, y)$, saying that the variable y is functionally dependent on (i.e. completely determined by) the variables $x_1, \dots, x_m$. Dependence logic is first-order logic together with such dependency atoms. As in Hodges' approach the semantics is compositionally defined in terms of sets of assignments, called *teams*.

Väänänen's approach has many advantages compared to the previous ones. It made the logical resoning about dependence mathematically much more transparent and led to a deeper understanding of the logical aspect of dependence and the expressive power of IF-logic and dependence logic.

Recent work by several authors has revealed that dependence is just one among many different properties that give rise to interesting logics based on team semantics. In [9] we have discussed the notion of independence (which is a much more delicate but also more powerful notion than dependence) and introduced independence logic. Galliani [7] and Engström [6] have studied several logics with team properties based on notions originating in database dependency theory.

We here give a survey, without proofs, on our work in [9] and [8] about logics of dependence and independence, and discuss also related results by Galliani [7] and by Kontinen

ICDT 2012, March 26–30, 2012, Berlin, Germany.
Copyright 2012 ACM 978-1-4503-0791-8/12/03 ...$10.00

and Väänänen [14, 15]. We want to show that the intuitive notions of dependence and independence can be treated as atomic statements in suitable logics (just like their cousin, identity). In this way, dependence and independence become logical notions with suitable axiomatizations, and they give rise to an emerging new logical theory.

2. DEPENDENCE AND INDEPENDENCE AS ATOMIC FORMULAE

The strongest form of dependence is functional dependence. This is the kind of dependence in which some given variables completely determine certain other variables, as surely as x and y determine $x + y$ and $x \cdot y$ in elementary arithmetic. The idea is that weaker forms of dependence can be understood in terms of the strongest. Following [16], functional dependence of y on $\overline{x}$ is denoted by the symbol $=(\overline{x}, y)$. By adopting the shorthand $=(\overline{x}, \overline{y})$ for $=(\overline{x}, y_1) \wedge \ldots \wedge =(\overline{x}, y_n)$ we get more general dependence atoms. Although there are many different intuitive meanings for $=(\overline{x}, \overline{y})$, such as *"$\overline{x}$ totally determines $\overline{y}$"* or *"$\overline{y}$ is a function of $\overline{x}$"*, the best way to understand the concept is to give it semantics. Let $\mathfrak{A}$ be a structure with universe A and let $\mathcal{V}$ be a set of variables. A team X with domain $\mathcal{V}$ and values in $\mathfrak{A}$ is a set of assignments $s : \mathcal{V} \to A$.

Definition 2.1. *A team X satisfies the dependency atom $=(\overline{x}, \overline{y})$, in symbols $\mathfrak{A} \models_X =(\overline{x}, \overline{y})$, if*

$$\forall s, s' \in X(s(\overline{x}) = s'(\overline{x}) \to s(\overline{y}) = s'(\overline{y})). \qquad (1)$$

Condition (1) is a universal statement. As a consequence it is closed downward, that is, if a team satisfies it, every subteam does. Further, every dependency atom is, for trivial reasons, satisfied by the empty team and by every singleton team $\{s\}$.

A long time ago, functional dependence has been studied in database theory and some basic properties, called *Armstrong's Axioms* have been isolated [2]. These axioms state the following properties of $=(\overline{x}, \overline{y})$:

(1) $=(\overline{x}, \overline{x})$. Anything is functionally dependent of itself.

(2) If $=(\overline{x}, \overline{y})$ and $\overline{x} \subseteq \overline{z}$, then $=(\overline{z}, \overline{y})$. Functional dependence is preserved by increasing input data.

(3) If $=(\overline{x}, \overline{y})$, $\overline{z}$ is a permutation of $\overline{x}$, and $\overline{u}$ is a permutation of $\overline{y}$, then $=(\overline{z}, \overline{u})$. Functional dependence does not look at the order of the variables.

(4) If $=(\overline{x}, \overline{y})$ and $=(\overline{y}, \overline{z})$, then $=(\overline{x}, \overline{z})$. Functional dependences can be transitively composed.

These rules completely describe the behaviour of $=(\overline{x}, \overline{y})$ in the following sense: If T is a finite set of dependence atoms of the form $=(\overline{x}, \overline{y})$ for various $\overline{x}$ and $\overline{y}$, then $=(\overline{x}, \overline{y})$ follows from T according to the above rules if and only if every team that satisfies T also satisfies $=(\overline{x}, \overline{y})$.

We shall now give the concept of independence a similar treatment. Independence is a much more subtle notion than dependence and is not just the absence of dependence. We start from the intuition that two variables x and y are independent if learning more about one does not convey any information whatsoever about the other. We thus oberve a kind of total lack of connection between them. More precisely, suppose we know a team X and we know that s is an assignment in X, but we have no further information about the values of s. Thus, for every variable z in the domain of X we just know that $s(z) \in \{s'(z) : s' \in X\}$. Now, independence of x and y in X means that learning the value $s(x)$ does not provide any additional information about the potential values of $s(y)$.

A different, but equivalent way to say this is that that values for (x, y) appear in all conceivable combinations: if values (a, b) and (a', b') occur for (x, y), then so do (a, b') and (a', b).

We discuss some classical scenarios where independence plays a role. Suppose balls of different sizes and masses are dropped from the Leaning Tower of Pisa in order to observe how size and mass influence the time of descent. In setting up such an experiment, one may want to make sure the following:

> *The size of the ball is independent of the mass of the ball.*

To satisfy this requirement one would vary the sizes and the masses freely so that if one mass is chosen for one size it would also have to occur for all the other sizes, and if one size is chosen for one mass it also appears with all other masses. This would eliminate any dependence between size and mass and the test would genuinely tell us something about the time of descent itself. We would then say that the size and the mass were made independent of each other in the strongest sense of the word.

Suppose we have data about tossing two coins and we want to state:

> *Whether one coin comes heads up is independent of whether the other coin comes heads up.*

To be convinced, we would look at the data and point out that all four possibilities occur. Of course, probability theory has its own concept of independence which however is in harmony with ours, only we do not pay attention to how many times a certain pattern occurs. In probability theory, roughly speaking, two random variables are independent if observing the other does not affect the (conditional) probability of the other. We could say the same without paying attention to probabilities as follows: two variables are independent if observing one does not restrict in any way what the value of the other is.

We next look at two examples of a seemingly different nature. When Galileo dropped balls of the same size from the Leaning Tower of Pisa he was able to observe:

> *The time of descent of an object is independent of its mass.*

Einstein stated in his theory of special relativity:

> *The speed of light is independent of the observer's state of motion.*

These are famous examples of independence where one of the variables is constant. The intuition that independence means that values appear in all conceivable combinations, or that learning a value of one does not tell us anything about the other, is satisfied here as well (although in a somewhat trivial way). So we should accept that one form of total independence is when one of the variables is a constant. Another feature of this strong form of independence is symmetry. There are weaker forms of independence where symmetry is not present.

Let us now introduce the semantics of the independence atom $\overline{x}\perp\overline{y}$:

Definition 2.2. *A team X satisfies the atomic formula $\overline{x}\perp\overline{y}$ if*

$$\forall s, s' \in X \exists s'' \in X(s''(\overline{x}) = s(\overline{x}) \wedge s''(\overline{y}) = s'(\overline{y})). \qquad (2)$$

We immediately observe that a constant variable is independent of every other variable, including itself. We can also immediately observe the symmetry of independence, because the criterion (2) is symmetrical. Independence can be axiomatized in a similar way as dependence, using the following axioms.

(1) If $\overline{x}\perp\overline{y}$, then $\overline{y}\perp\overline{x}$ (Symmetry Rule).

(2) If $\overline{x}\perp\overline{y}$, and $\overline{z} \subseteq \overline{x}$, then $\overline{z}\perp\overline{y}$.

(3) If $\overline{x}\perp\overline{y}$, $\overline{z}$ is a permutation of $\overline{x}$, and $\overline{u}$ is a permutation of $\overline{y}$, then $\overline{z}\perp\overline{u}$.

(4) If $\overline{x}\perp\overline{y}$ and $\overline{xy}\perp\overline{z}$, then $\overline{x}\perp\overline{yz}$.

The independence atom $\overline{x}\perp\overline{y}$ actually is a special case of the more general notion

$$\overline{x} \perp_{\overline{z}} \overline{y}$$

the intuitive meaning of which is that the variables $\overline{y}$ are totally independent of the variables $\overline{x}$ when the variables $\overline{z}$ are kept fixed.

Suppose objects of different forms (balls, pins, etc), different sizes and different masses are dropped from the Leaning Tower of Pisa in order to observe how the form, size and mass influence the time of descent. One may want to make sure that in this test:

> *For any fixed form, the size of the object is independent of the mass of the object.*

To make sure of this, one would vary for each form separately the sizes and the masses freely so that if one mass is chosen in that form for one size it would be also be chosen in that form for all the other sizes, and so on. We would then say that the size and the mass were made independent of each other, given the form, in the strongest sense of the word.

We now give mathematical content to $\overline{x} \perp_{\overline{z}} \overline{y}$:

Definition 2.3. *A team X satisfies the atomic formula $\overline{x} \perp_{\overline{z}} \overline{y}$ if for all $s, s' \in X$ such that $s(\overline{z}) = s'(\overline{z})$ there exists $s'' \in X$ such that $s''(\overline{z}) = s(\overline{z})$, $s''(\overline{x}) = s(\overline{x})$, and $s''(\overline{y}) = s'(\overline{y})$.*

Lemma 2.4. *(1) $=(\overline{x}, \overline{y})$ logically implies $\overline{y} \perp_{\overline{x}} \overline{z}$.*

(2) $\overline{y} \perp_{\overline{x}} \overline{z}$ logically implies $=(\overline{x}, \overline{y} \cap \overline{z})$.

(3) $=(\overline{x}, \overline{y}) \leftrightarrow \overline{y} \perp_{\overline{x}} \overline{y}$

So dependence is a special case of independence, when independence is defined in the more general form. This has the pleasant consequence that when we define *independence logic $\mathcal{I}$* by adding the atomic formulas $\overline{x} \perp_{\overline{z}} \overline{y}$ to first order logic, we automatically include all of dependence logic.

We get the following reformulation of (3):

Corollary 2.5. $\overline{y} \perp_{\overline{x}} \overline{y} \Rightarrow \overline{y} \perp_{\overline{x}} \overline{z}$ *(Constancy Rule)*

As above, we collect some axioms for $\overline{x} \perp_{\overline{z}} \overline{y}$:

(1) $\overline{x} \perp_{\overline{x}} \overline{y}$ (Reflexivity Rule)

(2) $\overline{z} \perp_{\overline{x}} \overline{y} \Rightarrow \overline{y} \perp_{\overline{x}} \overline{z}$ (Symmetry Rule)

(3) $\vec{y}y' \perp_{\vec{x}} \vec{z}z' \Rightarrow \vec{y} \perp_{\vec{x}} \vec{z}$. (Weakening Rule)

(4) If $\vec{z'}$ is a permutation of $\overline{z}$, $\vec{x'}$ is a permutation of $\overline{x}$, $\vec{y'}$ is a permutation of $\overline{y}$, then $\overline{y} \perp_{\overline{x}} \overline{z} \Rightarrow \vec{y'} \perp_{\vec{x'}} \vec{z'}$. (Permutation Rule)

(5) $\overline{z} \perp_{\overline{x}} \overline{y} \Rightarrow \overline{yx} \perp_{\overline{x}} \overline{zx}$ (Fixed Parameter Rule)

(6) $\overline{x} \perp_{\overline{z}} \overline{y} \wedge \vec{u} \perp_{\vec{z}\vec{x}} \vec{y} \Rightarrow \overline{u} \perp_{\overline{z}} \overline{y}$. (First Transitivity Rule)

(7) $\overline{y} \perp_{\overline{z}} \overline{y} \wedge \overline{zx} \perp_{\overline{y}} \overline{u} \Rightarrow \overline{x} \perp_{\overline{z}} \overline{u}$ (Second Transitivity Rule)

Note that the Second Transitivity Rule gives by letting $\overline{u} = \overline{x}$:

$$\overline{y} \perp_{\overline{z}} \overline{y} \wedge \overline{x} \perp_{\overline{y}} \overline{x} \Rightarrow \overline{x} \perp_{\overline{z}} \overline{x},$$

which is the transitivity axiom of functional dependence. In fact Armstrong's Axioms are all derivable from the above rules.

There are of course many other atomic properties of teams that can be understood as dependency properties. Database dependency theory (see e.g. [1]) is one source of such properties. In fact, the independence atom discussed above is very closely related to the notion of *multivalued dependency* (see [6]). Also multivalued dependency can be used as an atom on teams, but one should take care to make the variables explicit, to make sure that the atom only depends on the variables actually appearing in it. Of specific interest are further properties known from dependency theory such as inclusion, exclusion, equiextension, etc.

Definition 2.6.

(1) A team X satisfies an inclusion atom *$\overline{x} \subset \overline{y}$ if for all $s \in X$ there is an $s' \in X$ with $s(\overline{x}) = s'(\overline{y})$.*

(2) A team X satisfies an exclusion atom *$\overline{x} \mid \overline{y}$ if for all $s, s' \in X$, $s(\overline{x}) \neq s'(\overline{y})$.*

(3) A team X satisfies an equiextension atom *$\overline{x} \bowtie \overline{y}$ if $\{s(\overline{x}) : s \in X\} = \{s(\overline{y}) : s \in X\}$.*

Results from dependency theory show that for these kinds of atoms and for combinations thereof, axiomatizations can been given.

3. LOGICS OF DEPENDENCE AND INDEPENDENCE

The atomic formulae stating dependence or independece properties can be combined with the common logical operators, such as connectives and quantifiers to obtain full-fledged logics for reasoning about dependence and independence. One aspect that makes these logics interesting and different from common logical systems such as first-order logic, modal logic, or second-order logic, is the requirement to evalute formulae against a set of assignments rather than a single assignment. We now explain this semantics, called team semantics.

In the sequel L is any logic, whose syntax extends first-order logic by atomic formulae on teams (such as dependence, independence, inclusion, exclusion or a combination thereof). We only admit atoms that are *local* in the sense that only the values assigned to variables that occur free in a formula are relevant for the truth of that formula. More formally, for every atom φ, every stucture $\mathfrak{A}$ and every team X we require that

$$\mathfrak{A} \models_X \varphi \quad \Longleftrightarrow \quad \mathfrak{A} \models_{X\restriction \mathrm{free}(\varphi)} \varphi.$$

Negation is, for good reasons (see the last section of this paper), used only in front of atomic formulae, i.e. formulae are always in negation normal form.

Let $\mathfrak{A}$ be a structure with universe A. An *assignment* (into $\mathfrak{A}$) is a map $s : \mathcal{V} \to A$ whose domain $\mathcal{V}$ is a set of variables. Given such an assignment s, a variable y, and an element $a \in A$ we write $s[y \mapsto a]$ for the assignment with domain $\mathcal{V} \cup \{y\}$ that updates s by mapping y to a.

A *team* is a set of assignments with the same domain. For a team X, a variable y, and a function $F : X \to \mathcal{P}(A)$, we write $X[y \mapsto F]$ for the set of all assignments $s[y \mapsto a]$ with $a \in F(s)$. Further we write $X[y \mapsto A]$ for the set of all assignments $s[y \mapsto a]$ with $a \in A$.

Team semantics for L defines whether a formula $\psi \in L$ is satisfied by a team X in a structure $\mathfrak{A}$, written $\mathfrak{A} \models_X \psi$. We always require that the domain of X contains all free variables of ψ.

We have already explained the semantics of the atomic formulae describing dependence, independence, and other basic properties of teams. Notice that all of these are trivially satisfied by the empty team. By definition, negated atoms of this kind are satisfied precisely by empty team. For instance, $\mathfrak{A} \models_X \neg{=}(x_1, \dots, x_m, y)$ if, and only if $X = \emptyset$. The further semantic rules are the following.

(1) If ψ is an atom $x = y$ or $Rx_1 \dots x_m$ or the negation of such an atom, then $\mathfrak{A} \models_X \psi$ if, and only if, $\mathfrak{A} \models_s \psi$ (in the sense of Tarski semantics) for all $s \in X$.

(2) $\mathfrak{A} \models_X (\varphi \wedge \vartheta)$ if, and only if, $\mathfrak{A} \models_X \varphi$ and $\mathfrak{A} \models_X \vartheta$.

(3) $\mathfrak{A} \models_X (\varphi \vee \vartheta)$ if, and only if, there exist teams Y, Z with $X = Y \cup Z$ such that $\mathfrak{A} \models_Y \varphi$ and $\mathfrak{A} \models_Z \vartheta$.

(4) $\mathfrak{A} \models_X \forall y\varphi$ if, and only if, $\mathfrak{A} \models_{X[y\mapsto A]} \varphi$.

(5) $\mathfrak{A} \models_X \exists y\varphi$ if, and only if, there is a map $F : X \to (\mathcal{P}(A) \setminus \emptyset)$ such that $\mathfrak{A} \models_{X[y\mapsto F]} \varphi$.

Notice that a disjunction is true in a team if that team can be split into subteams that satisfy the disjuncts. As a consequence $\varphi \vee \varphi$ is, in general, not equivalent to φ. Clause (5) giving semantics to existential quantifiers might seem surprising at first sight since it permits the choice of an arbitrary non-finite set of witnesses for an existentially quantified variable rather than a single witness (for each $s \in X$). What we use here has been called *lax semantics* in [7], as opposed to the more common *strict semantics.* For disjunctions (clause (3)) there is also a strict variant, requiring that the team X is split into *disjoint* subteams Y and Z. For first-order logic, and also for dependence logic the difference is immaterial since the two semantics are equivalent. However, this is no longer the case for stronger logics, in particular for independence logic. In these cases the lax semantics seems more appropriate since it preserves the locality principle that a formula should depend only on those variables that actually occur in it, whereas the strict semantics violates this principle. In game-theoretic terms the difference between strict and lax semantics corresponds to the difference between deterministic and nondeterministic strategies.

Notice that $\mathfrak{A} \models_\emptyset \psi$ holds for all formulae ψ.

If our formulae are just first-order, without dependence or independence atoms of any kind, then team semantics reduces to Tarski semantics. Indeed, it is easy to see that a first-order formula is satisfied by a team if, and only if, it is satisfied (in the sense of Tarski) by all assignments in it:

$$\begin{aligned} \mathfrak{A} \models_X \psi \quad \Longleftrightarrow \quad & \mathfrak{A} \models_{\{s\}} \psi \text{ for all } s \in X \\ & \mathfrak{A} \models_s \psi \text{ for all } s \in X. \end{aligned}$$

This changes radically when the formulae make use of dependence or independence atoms.

4. EXPRESSIVE POWER

Let us first consider dependence logic $\mathcal{D}$, the extension of first-order logic by dependence atoms $=(x_1, \dots x_m, y)$. A first observation is that the semantics of dependence logic is downwards closed for teams.

PROPOSITION 4.1 (DOWNWARDS CLOSURE). *For all* $\mathfrak{A}$, *all formulae* $\psi \in \mathcal{D}$ *and all teams* $Y \subseteq X$, *we have*

$$\mathfrak{A} \models_X \psi \quad \Longrightarrow \quad \mathfrak{A} \models_Y \psi.$$

We say that a structure $\mathfrak{A}$ is a model of a *sentence* $\psi \in \mathcal{D}$ if $\mathfrak{A} \models_{\{\emptyset\}} \psi$, i.e. if ψ is satisfied by the team that just contains the empty assignment. We thus can directly compare the expressive power of sentences of dependence logic with sentences of classical logics with Tarski semantics. It is not difficult to see that in this sense, dependence logic is equivalent to existential second-order logic Σ^1_1 (see [16]) and thus, by Fagin's Theorem expresses precisely those properties of finite structures that are in NP.

PROPOSITION 4.2. *For sentences,* $\mathcal{D} \equiv \Sigma^1_1$.

For formulae of dependence logic with free variables, such a direct comparison is not possible since dependence formulae are evaluated on teams and classical formulae on single assignments. However, a team X, with domain $\{x_1, \dots, x_k\}$ and values in A, can of course be represented by a relation $\mathrm{rel}(X) \subseteq A^k$, defined by $\mathrm{rel}(X) = \{(s(x_1), \dots, s(x_k)) : s \in X\}$. A formula ψ with vocabulary τ and free variables $x_1, \dots, x_k$ can then be translated into a Σ^1_1-sentence ψ^* of vocabulary $\tau \cup \{R\}$ such that, for every τ-structure $\mathfrak{A}$ and every team X

$$\mathfrak{A} \models_X \psi \quad \Longleftrightarrow \quad (\mathfrak{A}, \mathrm{rel}(X)) \models \psi^*.$$

Thus, on finite structures dependence logic can only express properties of teams that are in NP. The converse is not true since all properties of teams expressible in dependence logic are downwards closed (which of course need not be the case for arbitrary NP-properties). It was shown by Kontinen and Väänänen [14] that one can nevertheless precisely characterize the power of dependence formulae in terms of Σ^1_1-definability.

Theorem 4.3. *The expressive power of dependence logic is equivalent to the power of existential second-order sentences which are downwards monotone in the team predicate. Syntactically this means that dependence formulae are equivalent (on non-empty teams) to Σ^1_1-sentences in which the predicate for the team appears only negatively.*

An interesting special case of dependence atoms are those of form $=(y)$, expressing that $s(y)$ is constant, i.e. y takes the same value in all assignments $s \in X$. The fragment of dependence logic that only uses dependency atoms of this form is called *constancy logic* (see [7]). For sentences, constancy logic reduces to first-order logic, but this is not true for open formulae. Indeed, even the formula $=(x)$ cannot be equivalent to a first-order formula since its semantics does not reduce to Tarski semantics.

We next consider Independence Logic $\mathcal{I}$, the extension of first order logic by the new atomic formulas $\overline{y}\perp_{\overline{x}}\overline{z}$ for all sequences $\overline{x}, \overline{y}, \overline{z}$ of variables. On the level of sentences, independence logic is equivalent to Σ^1_1, and thus also equivalent to dependence logic. However, on the level of formulae, independence logic is strictly stronger than dependence logic. Indeed, any dependence atom $=(\overline{x}, \overline{y})$ is equivalent to the independence atom $\overline{y}\perp_{\overline{x}}\overline{y}$, but independence logic is not downwards closed, so a converse translation is not possible. It had been posed as an open problem in [9] to characterize the NP-properties of teams that correspond to formulae of independence logic. Very recently, Galliani [7] solved this problem by showing that actually, *all* NP-properties of teams can be expressed in independence logic. To do so Galliani has studied the logics obtained by adding other atomic properties such as inclusion, exclusion, and equiextension to FO.

The expressive power of these logics can be summarized as follows.

Theorem 4.4. *(1) First-order logic with inclusion atoms is incomparable to dependence logic and strictly contained in independence logic.*

(2) First-order logic with exclusion is equivalent to dependence logic

(3) First-order logic with equiextension atoms is equally expressive as FO with inclusion atoms.

(4) First-order logic with inclusion and exclusion has the same expressive power as independence logic. Moreover, both logics are equivalent to Σ^1_1.

5. MODEL-CHECKING GAMES AND COMPLEXITY

Let L be any extension of first-order logic (with team semantics) by a collection of atomic formulae on teams (such as dependence, independence, constancy, inclusion, exclusion, equiextension ...). We design model checking games for L. For every formula $\psi(\overline{x}) \in L$ (which we always assume to be in negation normal form), every structure $\mathfrak{A}$ and every team X with domain free(ψ) we define a game $\mathcal{G}(\mathfrak{A}, X, \psi)$ as follows.

Let $T(\psi)$ be the syntax tree of ψ; its nodes are the *occurrences* of the subformulae of ψ, with edges leading from any formula to its immediate subformulae, i.e. from $\varphi \vee \vartheta$ and $\varphi \wedge \vartheta$ to both φ and ϑ and from $\exists y\varphi$ and $\forall y\varphi$ to φ. The model-checking game $\mathcal{G}(\mathfrak{A}, X, \psi)$ is obtained by taking an appropriate product of $T(\psi)$ with assignments mapping variables to elements of $\mathfrak{A}$. The positions of the game are the pairs (φ, s) consisting of a node $\varphi \in T(\psi)$ and an assignment $s : \mathrm{free}(\varphi) \to A$. Verifier (Player 0) moves from positions associated with disjunctions and with formulae starting with an existential quantifier. From a position $(\varphi \vee \vartheta, s)$, she moves to either (φ, s') or (ϑ, s'') where s', s'' are the restrictions of s to the free variables of φ and ϑ, respectively. From a position $(\exists y\varphi, s)$, Verifier can move to any position $(\varphi, s[y \mapsto a])$, where a is an arbitrary element of A. Dually, Falsifier (Player 1) makes corresponding moves for conjunctions and universal quantifications. If φ is a literal then the position (φ, s) is terminal and attributed to none of the players.

Notice that the game tree, the rules for moves, and the set of plays are the same as in model checking games for first-order logic (in the usual sense, with Tarski semantics). However, there are some important differences.

First, in model-checking games for classical logics, it is not necessary to work with the syntax *tree*. Instead one can take a more compact representation by a directed acyclic graph (dag) that identifies different occurrences of the same subformula. For logics with team semantics it is relevant that we actually take the syntax tree, i.e., that we distinguish between different occurrences of the same subformula. Indeed, for instance, a formula $\varphi \vee \varphi$ is not equivalent to φ, and in its evaluation, different teams are typically attributed to the two occurrences of φ in $\varphi \vee \varphi$. A more relevant difference concerns the winning conditions and the associated strategies that we want to synthesize. The model checking games for logics with team semantics are not reachability games. In fact, winning or losing are not properties that can be attributed to terminal positions and, indeed, not even to single plays. Due to the underlying team semantics, and also due to the additional atomic formulae on teams, winning or losing is always a property of a strategy or of a set of plays, and not of a single play.

We can view a model-checking game as a structure of the form $\mathcal{G}(\mathfrak{A}, X, \psi) = (V, V_0, V_1, T, E)$ where V is the set of positions, V_σ is the set of positions where Player σ moves, T is the set of terminal positions (associated to literals), and E is the set of moves. In general, a nondeterministic (positional) winning strategy for Player 0 is a subgraph $S = (W, F) \subseteq (V, E)$ where W is the set of positions from which the strategy is winning (it need not be defined on other positions) and $F \subseteq E \cap (W \times W)$ is the set of moves that are consistent with the strategy. Beyond the obvious consistency requirements for strategies (see (1) and (2) below) we here introduce a third condition that is new and specific for team semantics. For that, we introduce the following notion. Given $S = (W, F)$ and a formula $\varphi \in T(\psi)$, the team associated with S and φ is

$$\text{Team}(S, \varphi) = \{s : (\varphi, s) \in W\}.$$

Informally the new condition (3) requires that every literal is satisfied by the team that the strategy associates with it.

Definition 5.1. *A* consistent winning strategy *for Verifier (Player 0) with winning region W in*

$$\mathcal{G}(\mathfrak{A}, \psi) = (V, V_0, V_1, T, E)$$

is a subgraph $S = (W, F) \subseteq (V, E)$ with $F \subseteq E \cap (W \times W)$ satifying the following three conditions:

(1) If $v \in W \cap V_0$, then vF is non-empty.

(2) If $v \in W \cap V_1$ then $vF = vE$.

(3) For every literal φ, we have that $\mathfrak{A} \models_{Team(S,\varphi)} \varphi$.

Recall that the empty team satisfies all formulae. If a literal φ has no occurrence $(\varphi, s) \in W$, then $\text{Team}(S, \varphi) = \emptyset$, and thus $\mathfrak{A} \models_{\text{Team}(S,\varphi)} \varphi$ is true for trivial reasons.

Notice that in the case where L is first-order logic (with team semantics, but without additional atoms), the third condition is equivalent to saying that $\mathfrak{A} \models_s \varphi$ for all literals φ and all assignments s with $(\varphi, s) \in W$. This is in harmony with the classical game-theoretic semantics for FO and reflects the fact that, for any first-order formula ψ, $\mathfrak{A} \models_X \psi$ if, and only if, Verifier has a winning strategy from *all* initial positions (ψ, s) with $s \in X$.

In fact, this generalizes beyond first-order logic.

Theorem 5.2. *For every structure $\mathfrak{A}$, every formula $\psi(\overline{x}) \in L$ and every team X with domain* free(ψ) *we have that $\mathfrak{A} \models_X \psi$ if, and only if, Player 0 has a consistent winning strategy $S = (W, F)$ for $\mathcal{G}(\mathfrak{A}, X, \psi)$ with $Team(S, \psi) = X$.*

For a proof, see [8].

We can find more abstract and purely combinatorial variants of such game-theoretic problems, abstracting away from logics with team semantics and model-checking problems, but focussing on winning strategies satisfying abstract consistency criteria.

We consider finite game graphs $\mathcal{G} = (V, V_0, V_1, I, T, E)$, with set of positions V, partioned as above into the sets V_0, V_1 and the set T of terminal positions, where E is the set of moves and I is the set of initial positions. Further, let $\text{Win} \subseteq \mathcal{P}(T)$ be a winning condition defining for each set $U \subseteq T$ of terminal position whether it is a winning set for Player 0. For algorithmic concerns, let us assume that it can be decided in polynomial time whether a given set $U \subseteq T$ belongs to Win.

Definition 5.1 of consistent winning strategies is then simplified and generalized as follows.

Definition 5.3. *A* consistent winning strategy *for Player 0 for a game $\mathcal{G} = (V, V_0, V_1, I, T, E)$ with winning condition* Win *is a subgraph $S = (W, F) \subseteq (V, E)$ with $F \subseteq E \cap (W \times W)$ satifying the following conditions:*

(1) If $v \in W \cap V_0$, then vF is non-empty.

(2) If $v \in W \cap V_1$ then $vF = vE$.

(3) $W \cap T \in \text{Win}$

(4) $I \subseteq W$.

Notice that item (4) requires a winning strategy to be winning from *all* initial positions.

Theorem 5.4. *The problem whether a given game graph $\mathcal{G}$ with an oracle for* Win *admits a consistent winning strategy for Player 0, is NP-complete.*

The *width* of a formula ψ is defined as the maximal number of free variables in subformulae of ψ, formally

$$\text{width}(\psi) := \max\{|\text{free}(\varphi)| : \varphi \in T(\psi)\}.$$

Notice that the size of a model checking game $\mathcal{G}(\mathfrak{A}, X, \psi)$ on a finite structure $\mathfrak{A}$ is bounded by $|T(\psi)| \cdot |A|^{\text{width}(\psi)}$.

Theorem 5.5. *Let L be any extension of first-order logic with team semantics by atomic formulae on teams that can be evaluated in polynomial time. Then the model-checking problem for L on finite structures is in* Nexptime. *For formulae of bounded width, the model-checking problem is in NP.*

In fact, with team semantics, the model-checking problem is Nexptime-complete already for relatively simple extensions of first-order logic. For first-order logic itself, it is Pspace complete, since without additional atoms, FO with team semantics reduces to FO with Tarski semantics. In particular, the model-checking for dependence logic is Nexptime complete, which can be proved by an encoding of an appropriate domino problems [8].

Theorem 5.6. *The problem to decide, given a finite structure $\mathfrak{A}$, a team X and a formula ψ in dependence logic, whether $\mathfrak{A} \models_X \psi$, is* Nexptime*-complete. This also holds when $\mathfrak{A}$ and X are fixed, in fact even in the case where $\mathfrak{A}$ is just the set $\{0, 1\}$ and $X = \{\emptyset\}$.*

It is not difficult to see that the same complexity results hold for independence logic, and logics using inclusion, exclusion, and/or equiextension atoms.

On the other side, constancy logic is a fragment of lower complexity.

Theorem 5.7. *The model checking problem for constancy logic is* Pspace*-complete.*

6. NEGATION

Negation is a nontrivial issue in logics of dependence and independence since we do not have the Law of Excluded Middle. This is reflected by the fact that the associated semantical games are usually not determined.

Given a formula $\psi \in L$ (where L is one of the logics considered above), let $\psi^\neg$ denote the formula in negation normal form that corresponds to the negation of ψ. For teams $X \neq \emptyset$, it cannot be the case that $\mathfrak{A} \models_X \psi$ and at the same time $\mathfrak{A} \models_X \psi^\neg$, but $\mathfrak{A} \not\models_X \psi$ does not imply that $\mathfrak{A} \models_X \psi^\neg$. We say that ψ is false for $\mathfrak{A}$ and X, if $\mathfrak{A} \models_X \psi^\neg$.

When just considering truth, one could describe the semantics of ψ in $\mathfrak{A}$ as the the set of all teams X with domain free(ψ) that satisfy the formula, i.e.

$$[\![\psi]\!]^{\mathfrak{A}} := \{X : \mathfrak{A} \models_X \psi\}.$$

However, when taking into account both truth and falsity, one should consider the pair $(\llbracket\psi\rrbracket^{\mathfrak{A}}, \llbracket\psi^{\neg}\rrbracket^{\mathfrak{A}})$ as the appropriate semantic value of ψ in $\mathfrak{A}$. This is also justified by results due to Burgess [4] and Kontinen and Väänänen [15] which show that for team semantics, negation is not really a semantic operation, contrary to disjunction, conjunction, and quantifiers. When we know $\llbracket\psi\rrbracket^{\mathfrak{A}}$ and $\llbracket\varphi\rrbracket^{\mathfrak{A}}$ we can easily compute $\llbracket\varphi \wedge \psi\rrbracket^{\mathfrak{A}}$ and $\llbracket\varphi \vee \psi\rrbracket^{\mathfrak{A}}$ (without even knowing the syntax of ψ and φ). Analogous observations hold for quantifiers. However, knowing $\llbracket\psi\rrbracket^{\mathfrak{A}}$ does not provide much knowledge about $\llbracket\psi^{\neg}\rrbracket^{\mathfrak{A}}$. Indeed, for any two formula ψ and φ of dependence logic that exclude each other (i.e. $\llbracket\psi\rrbracket^{\mathfrak{A}} \cap \llbracket\varphi\rrbracket^{\mathfrak{A}} = \{\emptyset\}$ on all $\mathfrak{A}$), there is formula ϑ such that $\llbracket\vartheta\rrbracket^{\mathfrak{A}} = \llbracket\psi\rrbracket^{\mathfrak{A}}$ and $\llbracket\vartheta^{\neg}\rrbracket^{\mathfrak{A}} = \llbracket\varphi\rrbracket^{\mathfrak{A}}$.

In what sense is the semantic value $(\llbracket\psi\rrbracket^{\mathfrak{A}}, \llbracket\psi^{\neg}\rrbracket^{\mathfrak{A}})$ described by the model-checking games? Notice that the game graphs of $\mathcal{G}(\mathfrak{A}, X, \psi)$ do not strongly depend on X. We can just as well account for all appropriate teams in a single game graph $\mathcal{G}(\mathfrak{A}, \psi)$.

Theorem 6.1. *In the game graph $\mathcal{G}(\mathfrak{A}, \psi)$, Player 0 has a consistent winning strategy S with $Team(S, \psi) = X$ precisely for the teams $X \in \llbracket\psi\rrbracket^{\mathfrak{A}}$ and Player 1 has a consistent winning strategy S' with $Team(S', \psi) = Y$ precisely for the teams $Y \in \llbracket\psi^{\neg}\rrbracket^{\mathfrak{A}}$.*

Notice that, $\psi^{\neg}$ is a formula in the same logic as ψ, and therefore equivalent also to a Σ^1_1-sentence, and, in general, not to one in Π^1_1. Further, the problem to check that a formula is false (for a given structure and a given team) is also in Nexptime (and in general not in Co-Nexptime).

7. REFERENCES

[1] S. Abiteboul, R. Hull, and V. Vianu, *Foundations of Databases*, Addison-Wesley, 1995.

[2] W. Armstrong, *Dependency structures of data base relationships*, Information Processing, 74 (1974).

[3] A. Blass and Y. Gurevich, *Henkin quantifiers and complete problems*, Annals of Pure and Applied Logic, 32 (1986), pp. 1–16.

[4] J. P. Burgess, *A remark on Henkin sentences and their contraries*, Notre Dame J. Formal Logic, 44 (2003), pp. 185–188.

[5] H. Enderton, *Finite partially ordered quantifiers*, Z. Math. Logik, 16 (1970), pp. 393–397.

[6] F. Engström, *Generalized quantifiers in dependence logic*. Draft, 2011.

[7] P. Galliani, *Inclusion and exclusion in team semantics — on some logics of imperfect information*, Annals of Pure and Applied Logic, 163 (2012), pp. 68–84.

[8] E. Grädel, *Model-checking games for logics of incomplete information*. Submitted for publication, 2012.

[9] E. Grädel and J.Väänänen, *Dependence and independence*, Studia Logica, (2012). To appear.

[10] L. Henkin, *Some remarks on infinitely long formulas*, in Infinitistic Methods, Warsaw, 1961, pp. 167–183.

[11] J. Hintikka and G. Sandu, *Informational independence as a semantical phenomenon*, in Studies in Logic and Foundations of Mathematics, vol. 126, North-Holland, 1989, pp. 571–589.

[12] W. Hodges, *Compositional semantics for a logic of imperfect information*, Logic Journal of IGPL, 5 (1997), pp. 539–563.

[13] W. Hodges, *Logics of imperfect information: Why sets of assignments?*, in Interactive Logic, J. van Benthem, B. Löwe, and D. Gabbay, eds., vol. 1 of Texts in Logic and Games, Amsterdam University Press, 2007, pp. 117–134.

[14] J. Kontinen and J. Väänänen, *On definability in dependence logic*, Journal of Logic, Language, and Information, 18 (2009), pp. 317–241.

[15] J. Kontinen and J. Väänänen, *A remark on negation in dependence logic*, Notre Dame Journal of Formal Logic, 52 (2011), pp. 55–65.

[16] J. Väänänen, *Dependence Logic*, Cambridge University Press, 2007.

[17] W. Walkoe, *Finite partially-ordered quantification*, Journal of Symbolic Logic, 35 (1970), pp. 535–555.