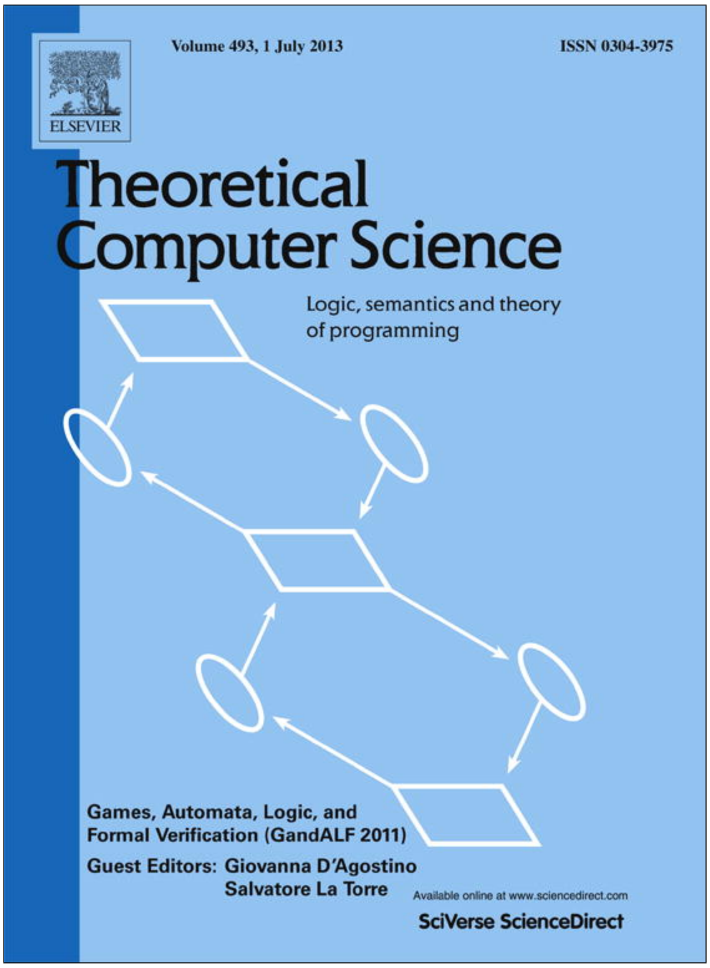
Volume 493, 1 July 2013
ISSN 0304-3975
ELSEVIER
Theoretical Computer Science
Logic, semantics and theory of programming
Games, Automata, Logic, and Formal Verification (GandALF 2011)
Guest Editors: Giovanna D'Agostino
Salvatore La Torre
Available online at www.sciencedirect.com
SciVerse ScienceDirect

Theoretical Computer Science 493 (2013) 2–14

Contents lists available at SciVerse ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

Model-checking games for logics of imperfect information

Erich Grädel *

Mathematische Grundlagen der Informatik, RWTH Aachen University, Germany

ARTICLE INFO

Keywords:
Model-checking games
Dependence logic
Independence logic
Team semantics
Second-order reachability games
Complexity

ABSTRACT

Logics of dependence and independence have semantics that, unlike Tarski semantics, are 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. We design model-checking games for logics with team semantics in a general and systematic way. The construction works for any extension of first-order logic by atomic formulae on teams, as long as certain natural conditions are observed which are satisfied by all team properties considered so far in the literature, including dependence, independence, constancy, inclusion, and exclusion.

The second-order features of team semantics are reflected by the notion of a consistent winning strategy which is also a second-order notion in the sense that it depends not on single plays but on the space of all plays that are compatible with the strategy. Beyond the application to logics with team semantics, we isolate an abstract, purely combinatorial definition of such games, which may be viewed as second-order reachability games, and study their algorithmic properties.

A number of examples are provided that show how logics with team semantics express familiar combinatorial problems in a somewhat unexpected way. Based on our games, we provide a complexity analysis of logics with team semantics.

1. Logics of imperfect information

Logics of imperfect information go back to the work of Henkin [10], Enderton [4], Walkoe [20], Blass and Gurevich [2], and others on partially ordered (or Henkin-) quantifiers, whose semantics can be naturally described in terms of games of imperfect information. A next step in this direction was 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. In fact it had repeatedly been 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 discovery of this kind of semantics is an achievement of independent interest, well beyond the application to IF logic. We believe that the full potential of this innovation has not yet been fully understood and appreciated. 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 [19] proposed a new approach to logics of imperfect information that changed the game significantly. Rather than stating dependencies or independencies as annotations of quantifiers, he proposed to consider dependence as an atomic formula, denoted $=(x_1, \ldots, x_m, y)$, saying that the variable y is functionally dependent on (i.e. completely determined by) the variables $x_1, \ldots, x_m$. Dependence logic is first-order logic together with such dependency atoms. As in

* Tel.: +49 241 8021730.
E-mail address: graedel@logic.rwth-aachen.de.

0304-3975/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2012.10.033

Hodges' approach the semantics is compositionally defined in terms of set of assignments, which he calls *teams*. Indeed a dependency statement $=(x_1, \dots, x_m, y)$ does not even make sense on a single assignment, but only on larger collection of data, given either by sets of assignments, or equivalently by a table or relation.

Väänänen's approach has many advantages compared to the previous ones. It made the logical reasoning 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. On the other side it perhaps has the disadvantage of making the game-theoretic aspects of the logics somewhat less intuitive. In [19] Väänänen discusses several variants of game-theoretic semantics for dependence logic. These either refer explicitly to the second-order features of the logic (positions in the game are given by teams rather than single assignments) and thus exponentially blow up the size of the games, or they need to go beyond the classical reachability games that are used for first-order logic (see e.g. [8]). In this paper we are going to discuss these model-checking games in a systematic way, not restricted to dependence logic but considering logics with team semantics in a general way.

Indeed, in the last years it has been shown by several authors 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 [6] and Engström [5] have studied several logics with team properties based on notions originating in database dependency theory.

We show that there is a uniform construction of model-checking games that works for all such logics. A natural condition that we require is the so-called locality principle, that a formula only depends on the free variables that actually occur in it. This principle goes almost without saying for classical logics with Tarski semantics, but it is a bit more delicate in the context of team semantics. The games that we introduce here can be seen as second-order reachability games. As in classical reachability (or safety) games the winning condition depends on terminal positions only (and not on histories of plays). But here the reachability condition that a winning strategy has to satisfy is a second-order condition which does not depend on single plays but on the space of all plays that are compatible with the strategy. Notice that second-order reachability games may well be of independent interest, beyond their applications to model-checking problems of logics with team semantics. We shall describe some basic properties and complexity issues for these games.

We shall also present a number of examples that show how logics with team semantics express familiar combinatorial problems in a somewhat unexpected way. Based on the model-checking games we shall further provide a complexity analysis for logics with team semantics. In particular we prove that the (combined complexity and expression complexity of the) model-checking problem for dependence and independence logic is NEXPTIME-complete, and NP-complete for formulae of bounded width.

Finally we shall discuss some variations of our games and the relationship to other models. We study the question under which conditions nondeterministic strategies are equivalent to deterministic ones. Further we discuss a known alternative construction of model-checking games which results in conceptually simpler games (reachability games of perfect information) which, however, require exponentially larger game graphs and thus do not lead to optimal complexity bounds. Finally, we consider the issue of imperfect information. Logics of dependence and independence are often called logics of imperfect information. We shall discuss the question, in what sense the model checking games for logics with team semantics are really games of imperfect information and pose the problem of how they can be related to more standard models of such games, such as the one presented by Reif [18].

One of the central results concerning games of imperfect information is a reduction, by a powerset construction, to games of perfect information of exponentially larger size. The problem of deciding winning positions for reachability (or safety) games of imperfect information is EXPTIME-complete. Thus, not only conceptually but also algorithmically, there are fundamental differences between our second-order reachability games, and in particular the model-checking games for logics with team semantics, on one side, and general (reachability) games of imperfect information on the other side. It is unclear whether the first can be understood as a special case of the latter. This is closely related to the question whether the notion of information really has a precise meaning in logics of imperfect information and how one can make this explicit.

1.1. Team semantics

Let $\mathfrak{A}$ be a structure of signature τ 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 a logic 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 and assume formulae to be in negation normal form. For first-order logic (FO) the 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 $\mathfrak{A} \models_{X[y\mapsto F]} \varphi$.

Remark. Clause (5) giving semantics to existential quantifiers might seem surprising at first sight since it permits the choice of an arbitrary non-empty 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 [6], 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 as given in the next section, 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.

As long as we stick to first-order logic, team semantics is uninteresting since it reduces to Tarski semantics. Indeed, it is easy to see that a formula is satisfied by a team if, and only if, it is satisfied (in the sense of Tarski) by all assignments in it:

$$\mathfrak{A} \models_X \psi \Leftrightarrow \mathfrak{A} \models_{\{s\}} \psi \quad \text{for all } s \in X$$
$$\mathfrak{A} \models_s \psi \quad \text{for all } s \in X.$$

This changes radically when we extend FO by atomic properties of teams describing dependencies (or independencies) among variables.

1.2. Dependence logic

The best studied such extension is Väänänen's dependence logic $\mathcal{D}$ that adds to FO (with team semantics) dependence atoms of form $=(x_1, \ldots, x_m, y)$, which express that the variable y is functionally dependent on $x_1, \ldots, x_n$. Such a statement does not make sense for a single assignment, but it has a precise meaning in a team of assignments. Formally, we add to the rules (1) to (5) given above the rule

(6) $\mathfrak{A} \models_X = (x_1, \ldots, x_m, y)$ if, and only if, for all assignments $s, s' \in X$ with $s(x_i) = s'(x_i)$ for $i = 1, \ldots, m$, we also have $s(y) = s'(y)$.

Negated dependency atoms (if allowed) are only satisfied by the empty team: $\mathfrak{A} \models_X \neg = (x_1, \ldots, x_m, y)$ if, and only if $X = \emptyset$.

Notice that $\mathfrak{A} \models_\emptyset \psi$ holds for all formulae ψ. Furthermore, one easily verifies that the semantics of dependence logic is downwards closed for teams.

Proposition 1.1 (*Downwards Closure*). *For all structures $\mathfrak{A}$, all formulae $\psi \in \mathcal{D}$ and all teams $Y \subseteq X$, we have*

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

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 [19]) and thus, by Fagin's Theorem expresses precisely those properties of finite structures that are in NP.

Proposition 1.2. *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 $\{x_1, \ldots, x_k\}$ and values in A, can be represented by a relation $\mathrm{rel}(X) \subseteq A^k$, defined by

$$\mathrm{rel}(X) = \{(s(x_1), \ldots, s(x_k)) : s \in X\}.$$

A formula ψ with vocabulary τ and free variables $x_1, \ldots, x_k$ 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

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

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 [15] that one can nevertheless precisely characterize the power of dependence formulae in terms of Σ^1_1-definability.

Theorem 1.3. *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 the 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 [6]). 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.

1.3. Independence logic

Independence is a much more subtle notion than dependence and is not just the absence of dependence. Intuitively two variables x and y are independent, denoted $x \perp y$, if acquiring more knowledge about one does not provide any additional knowledge about the other, which means 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).

To make this sufficiently general and precise Grädel and Väänänen [9] propose the general independence atom $\bar{y} \perp_{\bar{x}} \bar{z}$, for arbitrary tuples $\bar{x}, \bar{y}, \bar{z}$ of variables, which expresses that, for any fixed value of $\bar{x}$, the tuples $\bar{y}$ and $\bar{z}$ are independent in the following sense.

Definition 1.4. A team X satisfies $\bar{y} \perp_{\bar{x}} \bar{z}$ if, and only if, for all assignments $s, s' \in X$ such that $s(\bar{x}) = s'(\bar{x})$ there is an assignment $s'' \in X$ with $s''(\bar{x}) = s(\bar{x})$, $s''(\bar{y}) = s(\bar{y})$ and $s''(\bar{z}) = s'(\bar{z})$.

Now we define independence logic just as dependence logic $\mathcal{D}$ but using independence instead of dependence atoms.

Definition 1.5. Independence logic $\mathcal{I}$ is the extension of first order logic by the new atomic formulae $\bar{y} \perp_{\bar{x}} \bar{z}$ for all sequences $\bar{x}, \bar{y}, \bar{z}$ of variables. The negation sign $\neg$ is allowed in front of atomic formulae. The other logical operations are $\wedge$, $\vee$, $\exists$ and $\forall$. The semantics for independence atoms is as given in the definition above and in other cases exactly as for dependence logic.

We refer to [9] for a more detailed discussion and justification of independence logic. 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 $=(\bar{x}, y)$ is equivalent to the independence atom $y \perp_{\bar{x}} y$, but independence logic is not downwards closed, so a converse translation is not possible. It was posed as an open problem in [9] to characterize the NP-properties of teams that correspond to formulae of independence logic. Very recently, Galliani [6] solved this problem by showing that actually, *all* NP-properties of teams can be expressed in independence logic. To do so Galliani considered other atomic team properties such as inclusion and exclusion, and proved that together, they provide the same expressive power as independence.

1.4. Inclusion, exclusion, and all that

There are of course many other atomic properties of teams that can be used in logics with team semantics. Database dependency theory (see [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 [5]). 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 1.6. (1) A team X satisfies an *inclusion atom* $\bar{x} \subset \bar{y}$ if for all $s \in X$ there is an $s' \in X$ with $s(\bar{x}) = s'(\bar{y})$.
(2) A team X satisfies an *exclusion atom* $\bar{x} \mid \bar{y}$ if for all $s, s' \in X$, $s(\bar{x}) \neq s(\bar{y})$.
(3) A team X satisfies an *equiextension atom* $\bar{x} \bowtie \bar{y}$ if $\{s(\bar{x}) : s \in X\} = \{s(\bar{y}) : s \in X\}$.

Galliani [6] has studied the logics obtained by adding one or several of these atomic properties to FO, and has established the following results.

- First-order logic with inclusion atoms is incomparable to dependence logic and strictly contained in independence logic.
- First-order logic with exclusion is equivalent to dependence logic.
- First-order logic with equiextension atoms is equally expressive as FO with inclusion atoms.
- First-order logic with inclusion and exclusion has the same expressive power as independence logic. Moreover, both logics are equivalent to Σ^1_1.

There are many other conceivable properties of teams that one might add, as atomic formulae, to first-order logic. We will study here a uniform way to provide game-theoretic semantics for such logics, by a construction that does not depend on the particular atoms that are used.

2. Games for logics with team semantics

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, equi-extension etc.). 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 structure $\mathfrak{A}$ and every team X we require that

$$\mathfrak{A} \models_X \varphi \Leftrightarrow \mathfrak{A} \models_{X\upharpoonright \mathrm{free}(\varphi)} \varphi.$$

We design model checking games for L. For every formula $\psi(\bar{x}) \in L$ (which we always assume to be in negation normal form) and every structure $\mathfrak{A}$ we define a game $\mathcal{G}(\mathfrak{A}, \psi)$ 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 leaves of the tree are the nodes associated to literals. Notice that for logics with team semantics it is relevant that we actually take the syntax *tree* and not the associated dag, i.e., 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 $\varphi \vee \varphi$.

The model-checking game $\mathcal{G}(\mathfrak{A}, \psi)$ is obtained by taking an appropriate product of $T(\psi)$ with the set of assignments mapping variables to elements of $\mathfrak{A}$. More precisely, 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); see e.g. [8]. However, there are important differences concerning the winning conditions. The model checking games for logics with team semantics are not reachability games in the usual sense. 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 $\mathcal{G}(\mathfrak{A}, \psi) = (V, V_0, V_1, T, E)$ 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) strategy for Player 0 is a subgraph $S = (W, F) \subseteq (V, E)$ where W is the set of nodes 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), (2) and (3) below) we here introduce a further 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

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

Informally the new condition (4) requires that every literal is satisfied by the team that the strategy associates with it.

Definition 2.1. 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 $F \subseteq E \cap (W \times W)$ satisfying the following three conditions.

(1) Every node $v \in W$ is either a root, or has a predecessor $u \in W$ with $(u, v) \in F$.
(2) If $v \in W \cap V_0$, then vF is non-empty.
(3) If $v \in W \cap V_1$ then $vF = vE$.
(4) For every terminal position $v = (\varphi, s) \in W$ it is required that

$$\mathfrak{A} \models_{\mathrm{Team}(S,\varphi)} \varphi.$$

Remark. An equivalent formulation for item (4) would be the following.

(4)′ For every literal φ,

$$\mathfrak{A} \models_{\mathrm{Team}(S,\varphi)} \varphi.$$

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

Notice that in the case where L is first-order logic (with team semantics, but without additional atoms), condition (4) 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 $s \in X$.

We claim that this generalizes beyond first-order logic.

Theorem 2.2. *For every structure $\mathfrak{A}$, every formula $\psi(\bar{x}) \in L$ and every team X with domain* $\mathrm{free}(\psi)$ *we have that $\mathfrak{A} \models_X \psi$ if, and only if, Player* 0 *has a consistent winning strategy $S = (W, F)$ for $\mathcal{G}(\mathfrak{A}, \psi)$ with* $\mathrm{Team}(S, \psi) = X$.

Proof. We proceed by induction on ψ. First, let ψ be a literal. The game $\mathcal{G}(\mathfrak{A}, \psi)$ is just the set of isolated nodes (ψ, s) for all possible assignments s. If $\mathfrak{A} \models_X \psi$ then let $W_\psi = \{(\psi, s) : s \in X\}$ and $F_\psi = \emptyset$. Clearly $S_\varphi = (W_\varphi, F_\varphi)$ is a consistent winning strategy in $\mathcal{G}(\mathfrak{A}, \psi)$ with $\mathrm{Team}(S_\psi, \psi) = X$. If $\mathfrak{A} \not\models_X \psi$ then for any consistent winning strategy S with $\mathfrak{A} \models_{\mathrm{Team}(S,\psi)} \psi$ it must be the case that $\mathrm{Team}(S, \psi) \neq X$.

Next suppose that $\psi = \eta \vee \vartheta$. If $\mathfrak{A} \models_X \eta \vee \vartheta$ then there exist teams Y, Z with $X = Y \cup Z$ such that $\mathfrak{A} \models_Y \eta$ and $\mathfrak{A} \models_Z \vartheta$. By induction hypothesis there are consistent winning strategies $S_\eta = (W_\eta, F_\eta)$ in $\mathcal{G}(\mathfrak{A}, \eta)$ and $S_\vartheta = (W_\vartheta, F_\vartheta)$ in $\mathcal{G}(\mathfrak{A}, \vartheta)$ with $\mathrm{Team}(S_\eta, \eta) = Y$ and $\mathrm{Team}(S_\vartheta, \vartheta) = Z$. We obtain a consistent winning strategy $S_\psi = (W_\psi, F_\psi)$ in $\mathcal{G}(\mathfrak{A}, \psi)$ by $W_\psi := W_\eta \cup W_\vartheta \cup \{(\varphi, s) : s \in X\}$ and $F_\varphi := F_\eta \cup F_\vartheta \cup \{\langle(\psi, s)(\eta, s')\rangle : s \in Y, s' = s \restriction_{\mathrm{free}(\eta)}\} \cup \{\langle(\psi, s)(\vartheta, s')\rangle : s \in Z, s' = s \restriction_{\mathrm{free}(\vartheta)}\}$. Obviously $\mathrm{Team}(S_\psi, \psi) = X$ and since $X = Y \cup Z$ the strategy S_ψ admits, from every point $(\psi, s) \in W_\psi$ at least one edge to either (η, s') or (ϑ, s'). Conversely, every consistent winning strategy $S_\psi = (W_\psi, F_\psi)$ for Player 0 with $\mathrm{Team}(S_\varphi, \varphi) = X$ induces a decomposition $X = Y \cup Z$ where Y contains those $s \in X$ such that F_ψ admits a move from (ψ, s) to $(\eta, s \restriction_{\mathrm{free}(\eta)})$ and analogously for Z and ϑ. By induction hypothesis it follows that $\mathfrak{A} \models_Y \eta$ and $\mathfrak{A} \models_Z \vartheta$ and therefore $\mathfrak{A} \models_X \psi$.

The arguments for $\psi = \eta \wedge \vartheta$ are analogous (and in fact even simpler).

Let us now consider formulae $\psi = \exists y \varphi$. If $\mathfrak{A} \models_X \psi$ then there is a function $F : X \to (\mathcal{P}(A) \setminus \{\emptyset\})$ such that $\mathfrak{A} \models_{X[y \mapsto F]} \varphi$. By induction hypothesis, Player 0 has a consistent winning strategy $S_\varphi = (W_\varphi, F_\varphi)$ with $\mathrm{Team}(S_\varphi, \varphi) = X[y \mapsto F]$. We obtain a consistent winning strategy $S_\psi = (W_\psi, F_\psi)$ by setting $W_\psi := W_\varphi \cup \{(\psi, s) : s \in X\}$ and $F_\psi = F_\varphi \cup \{(s, s[y \mapsto a]) : s \in X, a \in F(s)\}$. Obviously, $\mathrm{Team}(S_\psi, \psi) = X$. Conversely, a consistent winning strategy $S_\psi = (W_\psi, F_\psi)$ with $\mathrm{Team}(S_\psi, \psi) = X$ requires that from every node (ψ, s) with $s \in X$ the set $(\psi, s)F_\psi$ of admissible successor nodes is non-empty. Let $F(s) := \{a \in A : (\varphi, s[y \mapsto a]) \in (\psi, s)F_\psi\}$. By induction hypothesis $\mathfrak{A} \models_{X[y \mapsto F]} \varphi$ and hence $\mathfrak{A} \models_X \psi$.

Again the arguments for formulae $\forall y \vartheta$ are analogous. □

3. Examples

It takes a bit of time to get acquainted with and develop an intuition for logics with team semantics. We describe here a few examples of relatively simple formulae, that describe well-known problems in a somewhat unexpected way. At the same time, we discuss the consistent winning strategies in the associated model-checking games.

3.1. Counting

As shown by Kontinen and Väänänen [15] there is, for every k and every polynomial $p(x) \in \mathbb{N}[x]$ a formula $\psi(x_1, \ldots, x_k) \in \mathcal{D}$ such that for every finite set A and every team X with domain $\{x_1, \ldots, x_k\}$,

$$A \models_X \psi \Leftrightarrow |X| \leq p(|A|).$$

We write the polynomial as a sum of monomials: $p(x) = \sum_{i=1}^{n} x^{m(i)}$ where $m(i)$ are (not necessarily distinct) natural numbers. We then set $\psi(x_1, \ldots, x_k) = \bigvee_{i=1}^{n} \varphi_i$ where

$$\varphi_i : \exists y_1 \ldots \exists y_{m(i)} \bigwedge_{j=1}^{k} {=}(y_1, \ldots, y_{m(i)}, x_j).$$

A consistent winning strategy S for the game $\mathcal{G}(A, \psi)$ with $\mathrm{Team}(S, \psi) = X$ first provides a decomposition $X = X_1 \cup \cdots \cup X_n$ of X into subteams and then continues with strategies S_i such that $\mathrm{Team}(S_i, \varphi_i) = X_i$. These strategies then produce, for every assignment $s \in X_i$, (a nonempty set of) values $(a_1, \ldots, a_{m(i)}) \in A^{m(i)}$ which completely determine s. This is possible if, and only if, $|X_i| \leq |A^{m(i)}|$. Hence a consistent winning strategy exists if, and only if $|X| \leq \sum_{i=1}^{n} |X_i| \leq \sum_{i+1}^{n} |A^{m(i)}| = p(|A|)$.

3.2. 3-colourability

An undirected graph G can be represented by a team X_G with attributes (i.e. variables) edge and node, consisting of all assignments $s : (\mathrm{edge}, \mathrm{node}) \mapsto (e, u)$ such that e is an edge of G and $u \in e$. The graph G is 3-colourable if, and only if, the associated team X_G satisfies the dependence formula

$$\psi(\mathrm{edge}, \mathrm{node}) := \exists \mathrm{colour}\big(({=}(\mathrm{colour}) \vee {=}(\mathrm{colour}) \vee {=}(\mathrm{colour})) \wedge$$
$$= (\mathrm{node}, \mathrm{colour}) \wedge {=}(\mathrm{edge}, \mathrm{colour}, \mathrm{node})\big).$$

To see this, consider the model-checking game. A consistent winning strategy $S = (W, F)$ with $\mathrm{Team}(S, \psi) = X_G$ requires to chose, for every assignment $s : (\mathrm{edge}, \mathrm{node}) \mapsto (e, u)$ a colour $c(s)$. The first conjunct in the formula imposes that this function takes three values: X_G is split into three subsets on each of which c is a constant. The second conjunct imposes that c takes the same value on all assignments with the same node, i.e. the colour of (e, u) only depends on the node u. The final conjunct requires that the edge and the colour determine the node, i.e. a different colour is assigned to (e, u) and (e, v) for every edge $e = \{u, v\}$. Altogether a consistent winning strategy with team X_G exists if, and only if, G admits a correct colouring with three colours.

3.3. The dependence formula for 3SAT

It has been shown by Jarmo Kontinen [14] that 3SAT can be expressed by a very simple formula of dependence logic, which is just a disjunction of three dependence atoms.

We present an instance $\varphi = \bigwedge_{i=1}^{m}(X_{i_1} \vee X_{i_2} \vee X_{i_3})$ of 3-SAT by a team

$$Z_\varphi = \{(i, j, X, \sigma) : \text{in clause } i \text{ at position } j\text{,the variable } X \text{ appears with parity } \sigma\}.$$

For instance, the formula $\varphi = (X_1 \vee \neg X_2 \vee X_3) \wedge (X_2 \vee X_4 \vee \neg X_5)$, is described by the team

clause	position	variable	parity
1	1	X_1	$+$
1	2	X_2	$-$
1	3	X_3	$+$
2	1	X_2	$+$
2	2	X_4	$+$
2	3	X_5	$-$

Proposition 3.1. *A propositional 3CNF-formula φ is satisfiable if, and only if, the associated team Z_φ is a model of*

$$=(\text{clause, position}) \vee =(\text{clause, position}) \vee =(\text{variable, parity}).$$

We consider the associated model checking game. If we write the dependence formula representing 3SAT as $\psi = \vartheta_1 \vee \vartheta_2 \vee \vartheta_3$ (where both ϑ_1 and ϑ_2 are $=$(clause,position) and ϑ_3 is $=$(variable,parity)) then the game defined ψ is a forest that contains, for each assignment s, a tree with root (ψ, s) from which three edges point to (ϑ_1, s_{cp}), (ϑ_2, s_{cp}) and $(\vartheta_3, s_{v\sigma})$, where s_{cp} and $s_{v\sigma}$ are the restrictions of s to {clause, position} and to {variable, parity}, respectively.

A consistent winning strategy $S = (W, F)$ for Player 0 in this game, with $\text{Team}(S, \psi) = Z_\varphi$, must select for each root (ψ, s) with $s \in Z_\varphi$ at least one outgoing edge. The team Z_φ contains, for each clause i, three assignments s_{i1}, s_{i2}, s_{i3} with $s_{ij}(\text{clause, position}) = (i, j)$. For any fixed i, the strategy S may permit a move to ϑ_1 from at most one of the three roots (ψ, s_{ij}), because otherwise $\text{Team}(S, \vartheta_1)$ would not satisfy ϑ_1. The same is true for ϑ_2. Hence S must permit, for each clause i, at least one move from a root (ψ, s_{ij}) to ϑ_3. This means that S selects, for each clause i, at least one of the three literals occurring in that clause. Further it must be the case that $\text{Team}(S, \vartheta_3) \models \vartheta_3$. But this means that in the set of selected literals, each variable appears with a fixed parity, or to put it differently, a consistent winning strategy actually defines a satisfying assignment for φ.

Remark. It should be noted that the dependence formula given by Jarmo Kontinen describes 3SAT correctly only for 3CNF-formulae with *precisely* three literals per clause (which is course no significant restriction). Indeed, formulae in dependence logic are closed under subteams. Removing an assignment from the team Z_φ corresponds to removing one occurrence of a literal in some clause of φ. While removing an assignment from Z_φ cannot change the truth value of a dependence formula from true to false, removing a literal can make a satisfiable 3CNF-formula unsatisfiable.

3.4. Independence versus Henkin quantifiers

As pointed out in [9], independence atoms can also be used to describe semantics of Henkin quantifiers. We illustrate this with the simple formula

$$\varphi := \begin{pmatrix} \forall x & \exists y \\ \forall u & \exists v \end{pmatrix} Pxyuv.$$

Semantics, either in terms of imperfect information games or, equivalently, in terms of Skolem functions give this formula, on a structure $\mathfrak{A} = (A, P)$ the meaning that there exist functions $f, g : A \to A$ such that $\mathfrak{A} \models P(a, fa, c, gc)$ for all $a, c \in A$.

Now, consider

$$\psi := \forall x \exists y \forall u \exists v (xy \bot_u v \wedge Pxyuv).$$

By our game-theoretic semantics this sentence is true on $\mathfrak{A} = (A, P)$ if there exist functions $F : A \to \mathcal{P}(A) \setminus \{\emptyset\}$ and $G : A \times A \times A \to \mathcal{P}(A) \setminus \{\emptyset\}$ such that $xy \bot_u v$ and $Pxyuv$ are satisfied by the team X_{FG} consisting of all assignments $(x, y, u, v) \mapsto (a, b, c, d)$ with $b \in F(a)$ and $d \in G(a, b, c)$.

We claim that the two formulae are equivalent. From Skolem functions f, g for φ we immediately get a consistent winning strategy for ψ, witnessed by $F(a) := \{fa\}$ and $G(a, b, c) := \{gc\}$. The teams X_{FG} then consists of all assignments (a, fa, c, gc) which clearly satisfies $Pxyuv$ and also satisfies the independence atom $xy \bot_u v$ since v is constant for any fixed value for u.

The converse is slightly more subtle. Given a consistent winning strategy for ψ on $\mathfrak{A}$, witnessed by F and G, the Skolem functions for φ are defined using a choice function. Fix a choice function ε on $\mathcal{P}(A)$ (assigning to every nonempty subset of A one of its members). Then let $fa := \varepsilon F(a)$ and $gc := \varepsilon(\bigcup_{a \in A} G(a, fa, c))$. We claim that $\mathfrak{A} \models P(a, fa, c, gc)$ for all $a, c \in A$. To prove this it suffices to show that, for all a, c, the assignment (a, fa, c, gc) belongs to X_{FG}. Given a, c, there exists some

$a' \in A$ such that $gc \in G(a', fa', c)$; hence X_{FG} contains the assignment (a', fa', c, gc). Further X_{FG} also contains all assignments (a, fa, c, d) where $d \in G(a, fa, c)$. Using the independence atom we infer that X_{FG} also contains (a, fa, c, gc), as required.

Remark. If one adopts strict semantics for existential quantifiers, one can simplify the independence formula to $\forall x \exists y \forall u \exists v (x \perp_u v \wedge Pxyuv)$. However, with lax semantics this simpler formula is not equivalent to φ.

3.5. Kernels in directed graphs

Here is an example of an NP-complete problem that is not closed under subteams and a way to express it using dependence, inclusion and exclusion.

We represent a directed graph by a team E with attributes (source, target).

A kernel of a directed graph is a set K of nodes such that no edges go from K to K and every node outside K is dominated by a node in K (there is an edge from K to it). The existence of a kernel is an NP-complete problem (GT57 in [7]). Notice that this problem is not downwards closed. For instance, the directed triangle has no kernel but making one of the three arcs bidirectional produces a kernel.

Consider the formula

$$\psi(\text{source}, \text{target}) = \exists c \forall x \exists y \forall x' \exists y' \big(=\!(c) \wedge =\!(x, y) \wedge =\!(x', y') \wedge (x = x' \to y = y') \wedge (\varphi \vee \vartheta)\big)$$

which expresses that there is constant c and two functions $f : x \mapsto y$ and $f' : x' \mapsto y'$, which are actually the same, such that the resulting team can be split into two subsets one of which satisfies φ and the other satisfies ϑ. The idea is that the kernel K of the graph is $f^{-1}(c)$ and the edge set (given by the team on source and target) is split into the team E_1 of edges originating in K and the team E_2 of edges originating in its complement. Thus we have to express that no edge in E_1 has its target in K, and that all nodes outside K are the target of an edge in E_1. This is achieved by setting

$$\varphi := (x = \text{source} \to (y = c \wedge (\text{source} \mid \text{target}) \wedge (y' = c \vee x' \subset \text{target})))$$
$$\vartheta := (x = \text{source} \to y \neq c).$$

We claim that, for any directed graph (V, E) we have

$$V \models_E \psi(\text{source}, \text{target}) \;\Leftrightarrow\; (V, E) \text{ has a kernel.}$$

Suppose that (V, E) has a kernel K. A consistent winning strategy for Verifier in the associated model checking game chooses a constant c and a function $f : V \to V$ such that $K = f^{-1}(c)$. The resulting team X (on the variables source, target, c, x, y, x', y') is split by the strategy into subteams $Y = \{s \in X : s(\text{source}) \in K)$ and $Z = \{s \in X : s(\text{source}) \notin K$. It is obvious that Z satisfies ϑ. To see that Y satisfies φ observe that the sets $\{s(\text{source}) : s \in Y\}$ and $\{s(\text{target} : s \in Y\}$ are disjoint, and that the set of possible values $s(x')$ splits into those with $s(x') \in K$ and hence $s'(y') = s(c)$ and those with $s(x') \notin K$. But since K is a kernel we have that every $s(x') \notin K$ is the target of an edge from K. Thus the inclusion atom is also satisfied.

Analogous arguments show that from any consistent winning strategy for $\mathcal{G}(V, E, \psi)$ one can indeed extract a kernel of (V, E).

4. Second-order reachability games

We now introduce an abstract and purely combinatorial variant of the games that we defined, abstracting away from logics with team semantics and model-checking problems, but focusing on winning strategies satisfying abstract consistency criteria. It turns out that the model-checking games for logics with team semantics are a special case of a second-order variant of reachability games.

Definition 4.1. A *second-order reachability game* is given by an acyclic game graph and a second-order reachability condition. Game graphs have the form $\mathcal{G} = (V, V_0, V_1, I, T, E)$, with set of positions V, partitioned 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. A second-order reachability condition is a collection Win $\subseteq \mathcal{P}(T)$ defining for each set $U \subseteq T$ of terminal positions whether it is a winning set for Player 0. For algorithmic concerns, let us assume that $\mathcal{G}$ is finite and that it can be decided in polynomial time whether a given set $U \subseteq T$ belongs to Win.

Definition 2.1 of consistent winning strategies is then simplified and generalized as follows.

Definition 4.2. A *consistent winning strategy* for Player 0 for a second-order reachability 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 $F \subseteq E \cap (W \times W)$ satisfying the following conditions.

(1) W is the set of nodes that are reachable from I via edges in F.
(2) If $v \in W \cap V_0$, then vF is non-empty.
(3) If $v \in W \cap V_1$ then $vF = vE$.
(4) $W \cap T \in$ Win.

In view of the remark after Definition 2.1 it is not difficult to see that this indeed generalizes the definitions for model-checking games. Notice that item (1) implies $I \subseteq W$ which means that a winning strategy must be winning from *all* initial positions.

Theorem 4.3. *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.*

Proof. Membership in NP is obvious. NP-hardness is also easy to prove, even for games where only Player 0 moves. Take for instance 3-colourability. With any graph $G = (V, E)$ we associate the second-order reachability game whose game graph is a forest, consisting of trees $T(v, e)$ for all edges $e \in E$ and all nodes $v \in e$. From the root of $T(v, e)$, Player 0 can move to the terminal nodes (v, e, i), for $i = 1, 2, 3$. The second-order reachability condition then specifies that a set U of terminal nodes is winning if, for any pair $(u, e, i), (v, f, j) \in U$ it holds that (1) if $u = v$ then $i = j$, and (2) if $e = f$ and $u \neq v$ then $i \neq j$. The set I of initial positions is the set of all roots. Clearly Player 0 has a consistent winning strategy if, and only if, the graph G is 3-colourable. □

Corollary 4.4. *The problem whether, for a given game $\mathcal{G}(\mathfrak{A}, \psi)$ and a team X, there exists a consistent winning strategy S with* $\text{Team}(S, \psi) = X$*, is NP-complete.*

5. Complexity of model-checking for logics with team semantics

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 $\mathcal{G}(\mathfrak{A}, \psi)$ on a finite structure $\mathfrak{A}$ is bounded by $|T(\psi)| \cdot |A|^{\text{width}(\psi)}$.

Theorem 5.1. *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.*

Proof. For a finite structure $\mathfrak{A}$ and a formula ψ, the model-checking game $\mathcal{G}(\mathfrak{A}, \psi)$ has exponential size. To determine whether Verifier has a consistent winning strategy is in NP with respect to the size of the game, and thus in NEXPTIME with respect to the inputs of the model-checking problem. However, for formulae of bounded width, the model-checking game has polynomial size. □

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. To prove this we proceed by expressing in dependence logic an appropriate domino problem. It is well-known that domino systems encode Turing machines in quite a direct way, and to prove, for instance, that a problem A is NEXPTIME hard, it suffices to show that, for every domino system $\mathfrak{D} = (D, H, V)$, one can reduce the problem, whether, given a word $w = w_0, \ldots, w_{n-1} \in D^n$, there exists a tiling $\tau : Z(2^n) \to D$ putting on every point (x, y) of the torus $Z(2^n) = \mathbb{Z}_{2^n} \times \mathbb{Z}_{2^n}$ (i.e. the square of size 2^n, where the upper and lower edges and the left and right edges are identified) a domino $d \in D$ such that horizontal and vertical adjacency conditions given by H and V are satisfied, and such that the initial condition given by $w \in D^n$ is satisfied. More precisely this means that

- if $\tau(x, y) = d$ and $\tau(x + 1, y) = d'$ then (d, d') *in* H,
- if $\tau(x, y) = d$ and $\tau(x, y + 1) = d'$ then $(d, d') \in V$,
- $\tau(i, 0) = w_i$ for $i = 0, \ldots, n - 1$.

Theorem 5.2. *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* $X = \{\emptyset\}$*.*

Proof. It suffices to show that, for any domino system $\mathfrak{D} = (D, H, V)$, we can provide an efficient reduction from a given initial condition $w \in D^n$ to a formula ψ_w such that $\mathfrak{D}$ admits a tiling of $Z(2^n)$ with initial condition w if, and only if $\mathfrak{A} \models_{\{\emptyset\}} \psi_w$. For simplicity of notation, we let $\mathfrak{A}$ be the structure with universe $D \cup \{0, 1\}$, with constants for $0, 1$ and all $d \in D$. By encoding elements of D by binary strings, and describing constants by existentially quantified elements and inequalities, this can easily be modified to $\mathfrak{A} = \{0, 1\}$. Points of $Z(2^n)$ are represented in binary by tuples $\bar{x} \in \{0, 1\}^n \times \{0, 1\}^n$. Obviously the conditions that two such tuples $\bar{x}$ and $\bar{x}'$ represent horizontally or vertically adjacent points can be described by quantifier-free first-order formula $h(\bar{x}, \bar{x}')$ and $v(\bar{x}, \bar{x}')$, and the same is true for the statements that $\bar{x}$ represents a point $(i, 0)$, for any fixed i. The tiling condition is then described by a dependence formula, where the dependence atom is used to represent the tiling function.

$$\psi_w := \forall\bar{x}\exists z\forall\bar{x}'\exists z'\big(=(\bar{x}', z') \wedge (\bar{x} = \bar{x}' \rightarrow z = z') \wedge$$
$$\left(h(\bar{x}, \bar{x}') \rightarrow \bigvee_{(d,d')\in H} (z = d \wedge z' = d')\right) \wedge$$
$$\left(v(\bar{x}, \bar{x}') \rightarrow \bigvee_{(d,d')\in V} (z = d \wedge z' = d')\right) \wedge$$
$$\bigwedge_{i=0}^{n} (\bar{x} = (i, 0) \rightarrow z = w_i)\big).$$

It is straightforward to verify that ψ_w expresses the desired tiling condition. □

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

Proof. Although constancy logic is, as far as open formulae are concerned, not equivalent to first-order logic, one can efficiently translate every formula $\psi(\bar{x})$ of constancy logic into one of the form

$$\exists z_1 \ldots \exists z_n \left(\bigwedge_{i=1}^{n} =(z_i) \wedge \varphi(\bar{x}, \bar{z})\right)$$

where $\varphi(\bar{x}, \bar{z})$ is a first-order formula. For a proof, see [6]. As a consequence the question whether $\mathfrak{A} \models_X \psi$ can be resolved by guessing values $c_1, \ldots, c_n$ for the variables $z_1, \ldots, z_n$ and deciding, for each assignment $s \in X$, whether $\mathfrak{A} \models \varphi(s(\bar{x}), \bar{c})$. Given that the model-checking problem for first-order logic is in Pspace, this procedure also runs in Pspace.

Pspace-hardness is clear, since constancy logic extends FO. □

6. The game for the opponent

Negation is a nontrivial issue in logics of imperfect information since we do not have the Law of Excluded Middle, and 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, via the usual dualities for propositional connectives and quantifiers.

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 $\mathfrak{A} \models_X \psi^{\neg}$.

How is the question whether ψ is false for $\mathfrak{A}$ and X, related to the model checking game $\mathcal{G}(\mathfrak{A}, \psi)$? Is the model-checking game sound also for falsity, rather than truth?

To answer this, we have, of course, to consider the winning strategies for the other player.

Definition 6.1. A *consistent winning strategy* for Falsifier (Player 1) in $\mathcal{G}(\mathfrak{A}, \psi) = (V, V_0, V_1, T, E)$ is a subgraph $S' = (W', F') \subseteq (V, E)$ with $F' \subseteq E \cap (W' \times W')$ satisfying the following conditions.

(1) Every node $v \in W'$ is either a root, or has a predecessor $u \in W$ with $(u, v) \in F'$.
(2) If $v \in W' \cap V_1$, then vF' is non-empty.
(3) If $v \in W' \cap V_0$ then $vF' = vE$.
(4) For every terminal position $v = (\varphi, s) \in W$ it is required that

$$\mathfrak{A} \models_{\mathrm{Team}(S',\varphi)} \neg\varphi.$$

By the same methods as in Section 2 one can prove that in this sense, the model-checking games are indeed sound also for falsity.

Theorem 6.2. *For every structure $\mathfrak{A}$, every formula $\psi(\bar{x}) \in L$ and every team X with domain* free(ψ) *we have that $\mathfrak{A} \models_X \psi^{\neg}$ if, and only if, Player* 1 *has a consistent winning strategy $S' = (W', F')$ for $\mathcal{G}(\mathfrak{A}, \psi)$ with* Team$(S', \psi) = X$.

When just considering truth, one could describe the semantics of ψ in $\mathfrak{A}$ as 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 $([\![\psi]\!]^{\mathfrak{A}}, [\![\psi^{\neg}]\!]^{\mathfrak{A}})$ as the appropriate semantic value of ψ in $\mathfrak{A}$. This is also justified by the observation of Kontinen and Väänänen [16] that for team semantics, negation is not really a semantic operation, contrary to disjunction, conjunction, and quantifiers. When we know $[\![\psi]\!]^{\mathfrak{A}}$ and $[\![\varphi]\!]^{\mathfrak{A}}$ we can easily compute $[\![\varphi\wedge\psi]\!]^{\mathfrak{A}}$ and $[\![\varphi\vee\psi]\!]^{\mathfrak{A}}$ (without even knowing the syntax of ψ and φ). Analogous observations hold for quantifiers. However, knowing $[\![\psi]\!]^{\mathfrak{A}}$ does not provide much knowledge about $[\![\psi^{\neg}]\!]^{\mathfrak{A}}$. Indeed, Kontinen and Väänänen prove that for any two formulae ψ and φ that exclude each other (i.e. $[\![\psi]\!]^{\mathfrak{A}} \cap [\![\varphi]\!]^{\mathfrak{A}} = \{\emptyset\}$) on all $\mathfrak{A}$, there is formula ϑ such that $[\![\vartheta]\!]^{\mathfrak{A}} = [\![\psi]\!]^{\mathfrak{A}}$ and $[\![\vartheta^{\neg}]\!]^{\mathfrak{A}} = [\![\varphi]\!]^{\mathfrak{A}}$.

In what sense is the semantic value $([\![\psi]\!]^{\mathfrak{A}}, [\![\psi^{\neg}]\!]^{\mathfrak{A}})$ described by the model-checking games?

Theorem 6.3. *In the game graph $\mathcal{G}(\mathfrak{A}, \psi)$, Player 0 has a consistent winning strategy S with* $\mathrm{Team}(S, \psi) = X$ *precisely for the teams $X \in [\![\psi]\!]^{\mathfrak{A}}$ and Player 1 has a consistent winning strategy S' with* $\mathrm{Team}(S', \psi) = Y$ *precisely for the teams $Y \in [\![\psi^{\neg}]\!]^{\mathfrak{A}}$.*

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. Variations of the games

In this final section, we discuss the relationship of our model-checking games and of second-order reachability games in general with some related model of games. First we consider the restriction to deterministic strategies. It is a natural question under which circumstances nondeterministic strategies are more powerful than deterministic ones. We shall present a necessary and sufficient criterion under which a second-order reachability condition guarantees that (for any game graph) nondeterministic wining strategies can be reduced to deterministic ones. Second, we shall briefly mention alternative model-checking games for logics with team semantics, which are games of perfect information but require exponentially larger game graphs than the ones that we introduced above. Finally we discuss the relationship of our second-order reachability games with more general models of imperfect information games and raise the question of how one can make the imperfect information in model-checking games explicit.

7.1. Deterministic versus nondeterministic strategies

Our notion of consistent winning strategies is nondeterministic. Of course one can also consider deterministic strategies, where condition (2) in Definition 4.2 is replaced by

(2)′ if $v \in W \cap V_0$ then $|vF| = 1$.

In most classical two-player games, deterministic strategies are no less powerful than nondeterministic ones. Hence the question arises whether this is also the case for the second-order reachability games studied here. As pointed out before this is closely related to the question whether the lax semantics for existential quantification (permitting to choose a set of values for an existentially quantified variable) is equivalent to the more common strict semantics where only one value is chosen. For instance, it is easy to see and well-known that the two semantics coincide for dependence logic.

A reason for this is that dependence logic is downwards closed. It is not difficult to prove that the downwards closure of the winning condition (which means that whenever $U \in \mathrm{Win}$ and $U' \subseteq U$, then also $U' \in \mathrm{Win}$) is a general sufficient condition for the equivalence of nondeterministic and deterministic winning strategies in a second-order reachability game.

Proposition 7.1. *Let* Win *be downwards closed. Then Player 0 has a consistent winning strategy for $\mathcal{G}$ and* Win *if, and only if, she has a deterministic one.*

Proof. Suppose that Player 0 has a consistent winning strategy $S = (W, F)$ for $\mathcal{G}$ and Win. Let f be a choice function assigning to every node $v \in W \cap V_0$ precisely one element of $F(v)$. We remove from F all edges (v, w) with $v \in W \cap V_0$ and $w \neq f(v)$; next we remove from (W, F) all nodes and edges that are no longer reachable from I via F-edges. The resulting strategy (W_0, F_0) is deterministic. Since $W_0 \cap T \subseteq W \cup T$ and since Win is downwards closed, it follows that $W_0 \cap T \in \mathrm{Win}$. Hence (W_0, F_0) is indeed a consistent winning strategy for Player 0. □

Is the downwards closure condition also necessary for eliminating nondeterminism from winning strategies?

The answer is no, but we can actually find a weaker condition which is necessary and sufficient for guaranteeing the possibility to eliminate nondeterministic strategies. We assume here that T is finite.

Definition 7.2. We say that a collection $\mathcal{F} \subseteq \mathcal{P}(T)$ has a split if there exist two subsets $U_1, U_2 \notin \mathcal{F}$ such that $U_1 \cup U_2 \in \mathcal{F}$.

Clearly, if $\mathcal{F}$ is downwards closed then it has no splits. The converse is not true.

Theorem 7.3. *Suppose that* $\mathrm{Win} \subseteq \mathcal{P}(T)$ *has no splits. Then Player 0 has a consistent winning strategy for a second-order reachability game given by a game-graph $\mathcal{G}$ and* Win *if, and only if, she has a deterministic one. Conversely, if* Win *has a split, then there exists a game graph $\mathcal{G}$ whose set of terminal nodes is T such that Player 0 has a consistent winning strategy for the second-order reachability game given by $\mathcal{G}$ and* Win*, but not a deterministic one.*

Proof. Suppose that Player 0 has a consistent winning strategy $S = (W, F)$ for $\mathcal{G}$ and Win. Take any node $v \in V_0 \cap W$ such that $|vF| > 1$. For any subset $X \subseteq F(v)$, let $S_{v,X} = (W_{v,X}, F_{v,X})$ be the strategy obtained by deleting from F all moves (v, y) with $y \notin X$, and then removing all nodes and edges that are no longer reachable from I.

We claim that for any decomposition $vF = X \cup Y$, we have that $W = W_{v,X} \cup W_{v,Y}$. It is clear that $W_{v,X} \cup W_{v,Y} \subseteq W$. For the reverse, notice that if $w \in W$ then there is a path along F-edges from I to w. If $w \notin W_{v,X}$ then every such path from I

to w goes through an edge (v, v') with $v' \notin X$. But then $v' \in Y$ and hence $w \in W_{v,Y}$. As a consequence, we also have that $W \cap T = (W_{v,X} \cap T) \cup (W_{v,Y} \cap T)$.

Based on this observation, we can transform S into a deterministic strategy as follows. Pick any node $v \in V_0 \cap W$ with $|vF| > 1$, and any successor $x \in vF$. Let $X = \{x\}$ and $Y = vF \setminus \{x\}$. If $W_{x,X} \cap T \in$ Win, then we update S to $S_{v,X}$ which is deterministic at node v. Otherwise we update S to $S_{v,Y} = (W_{v,Y}, F_{v,Y})$.

Since $W \cap T \in$ Win, and since $W \cap T = (W_{v,X} \cap T) \cup (W_{v,Y} \cap T)$ it follows that $W_{v,Y} \cap T \in$ Win since otherwise Win would have a split. Hence the updated strategy is still a consistent winning strategy. Further the updated strategy has now one successor less at v.

Hence iterating such modification steps will eventually produce a consistent winning strategy that is deterministic.

The converse is proved by a straightforward construction. If Win $\subseteq \mathcal{P}(T)$ has a split, i.e. a set $U = U_1 \cup U_2$ with $U \in$ Win, but $U_1, U_2 \notin$ Win, then let $\mathcal{G}$ be the game graph consisting of a tree whose root is in I, from which Player 0 has moves to v_1 and v_2; from v_1, Player 1 has moves to all terminal nodes in U_1 and from v_2 he has moves to all terminal nodes in U_2; finally add isolated terminal nodes for all $t \in T \setminus U$. Clearly Player 0 has a consistent winning strategy for this game, consisting of both possible moves from the root, but not a deterministic one. □

The following example, due to Galliani, shows the necessity of lax semantics and nondeterministic strategies if one considers logics with atoms that are not downwards closed and if one does not want to violate the locality principle that a formula should only depend on the variables that actually occur in it. Consider the formula $\psi = \exists x(y \subseteq x \wedge z \subseteq x)$ which expresses that the team under consideration can be extended by values for x in such a way that all values for y and z in the team occur also as values for x. Under lax semantics, this is obviously true for any team X since we can just assign to x all values occurring for y and z in the team. Under strict semantics, this formula is false for any team $X = \{s\}$ with $s(y) \neq s(z)$ which proves that strict semantics is different from lax semantics for inclusion logic. But observe that with strict semantics this formula violates the locality principle. Indeed, suppose that we have a team X' over the domain $\{y, z, u\}$ consisting of two assignments s, s' such that $s(y) = s'(y) \neq s(z) = s'(z)$ and $s(u) \neq s'(u)$. Clearly X' satisfies ψ for strict semantics, but if we restrict X' to the free variables y, z occurring in the formula, then the two assignments collapse to one and the formula becomes false.

7.2. Model-checking games with perfect information

Model-checking games for classical logics such as first-order logic, second-order logic, fixed-point logic, modal and temporal logics etc. are games of perfect information. In fact, also for dependence and independence logic, and indeed all logics with team semantics considered here, one can easily construct model checking games of perfect information. Such constructions can be directly read off the definition of the team semantics for these logics. In [19, Chapter 5.2] Väänänen presents an explicit such construction for dependence logic.

Of course, one has to pay a price to obtain a game with perfect information. Given a formula ψ and a structure $\mathfrak{A}$, the positions in a perfect-information model-checking game $\mathcal{PG}(\mathfrak{A}, \psi)$ are pairs (φ, X) where φ is a subformula of ψ and X is a team with domain free(φ) and values in $\mathfrak{A}$. The definition of the moves directly reflects the inductive definition for the team semantics. Perhaps the most interesting case concerns the moves associated with disjunctions, from positions of form $(\varphi \vee \vartheta, X)$. Here Player 0 has to provide a decomposition $X = Y \cup Z$, and then Player 1 determines whether the game proceeds from (φ, Y) or from (ϑ, Z). We leave the definition of the other details to the reader.

The advantage of such a construction is that we can stay with the familiar perfect information reachability games that we are used to and which we understand very well. However, there are several drawbacks. The first is that these games are not very natural and do not provide much insight. At positions associated with quantifiers, complex modifications of the current teams take place, which in some cases are not even governed by choices of the players. The games are closely related to the model-checking games that one would obtain from the translation of the given formula into existential second-order logic. The second, more serious, disadvantage is complexity. For a subformula φ with k free variables, we have to consider $2^{|A|^k}$ different positions. Even taking into account that reachability games with perfect information can be solved in linear time (with respect to the size of the game graph) we cannot obtain reasonable model-checking algorithms and optimal complexity bounds based on a direct translation into these games.

7.3. Games of imperfect information

Logics of dependence and independence, and generally logics with team semantics are often called logics of imperfect information. The associated model-checking games require, as we have seen, consistency (or uniformity) conditions on the winning strategies. We do not ask whether Player 0 has a winning strategy, but whether she has a consistent one, in order to get the correct truth definition. It is a common view shared by many people working in the field that games with such consistency conditions on the admissible strategies should be regarded as games of imperfect information. For instance, Väänänen writes:

> The uniformity requirement in effect makes the game into a non-determined game of imperfect information [19, p. 81].

However, the term "games of imperfect information" is often used in a somewhat different sense, for models of games where the information that the players have at a given position is made explicit, either in the form of information sets (sets of positions that are indistinguishable for a player) or in the form of a precise description of the private and public information at every state of the game.

A general and popular model for two-player graph games with imperfect information goes back to a paper by Reif [18] from the 1980s. The model is based on game graphs $G = (V, E)$ where every position is a quadruple $v = (i, p_0, p_1, c)$; here $i \in \{0, 1\}$ indicates the player who moves from v, p_0, p_1 are the private states of the two players at v, and c is the common state at v. The idea is that a player can see her own private state, the common state, and whose turn it is, but the private state of the opponent remains hidden. Thus a player cannot distinguish two positions that differ only in the private state of the opponent. One has to impose certain natural conditions on such graphs. The first requires that a move of one player cannot change the private state of the opponent, i.e. if $v = (i, p_0, p_1, c)$, $v' = (j, p'_0, p'_1, c')$ and $(v, v') \in E$, then $p'_{1-i} = p_{1-i}$. The second condition imposes that for any two indistinguishable positions v, v' of a player, the successors carry the same visible information for that player, i.e. if $v = (0, p_0, p_1, c)$, $v' = (0, p_0, p'_1, c)$, and there is a move $(v, w) \in E$ then there also exists a move $(v', w') \in E$ such that w and w' are indistinguishable for Player 0, and similarly for Player 1. There are many variations, often just other presentations, of this model with quite similar properties.

One of the central results concerning such games is that winning positions can be computed by a powerset construction, that reduces any game of imperfect information to a game of perfect information of exponentially larger size. Thus reachability and safety games of imperfect information (in the sense of Reif's model) can be solved in Exptime. In fact one cannot do essentially better as far as worst-case complexity results are concerned. Indeed, from Reif's paper [18] one can readily infer that the problem of deciding winning positions for reachability (or safety) games of imperfect information is Exptime-complete. For more detailed expositions, see e.g. [3,17].

Thus, not only conceptually but also algorithmically, there are fundamental differences between our second-order reachability games, and in particular the model-checking games for logics with team semantics on one side, and general (reachability) games of imperfect information on the other side.

It is an open question whether and how second-order reachability games can be viewed as, or admit a natural translation into, a special case of imperfect information games in the sense of Reif (or similar models). This is closely related to the question whether the notion of information really has a precise meaning in logics of imperfect information and how one can make this explicit. We believe that this is a fundamental question for the future research on logics with team semantics.

References

[1] S. Abiteboul, R. Hull, V. Vianu, Foundations of Databases, Addison-Wesley, 1995.
[2] A. Blass, Y. Gurevich, Henkin quantifiers and complete problems, Annals of Pure and Applied Logic 32 (1986) 1–16.
[3] K. Chatterjee, L. Doyen, T. A. Henzinger, J.-F. Raskin, Algorithms for omega-regular games with imperfect information, Logical Methods in Computer Science 3 (3).
[4] H. Enderton, Finite partially ordered quantifiers, Zeitschrift fur Mathematische Logik 16 (1970) 393–397.
[5] F. Engström, Generalized quantifiers in dependence logic, Journal of Logic, Language, and Information.
[6] P. Galliani, Inclusion and exclusion in team semantics—On some logics of imperfect information, Annals of Pure and Applied Logic 163 (2012) 68–84.
[7] M. Garey, D. Johnson, Computers and Intractability. A Guide to the Theory of NP-Completeness, Freeman, 1979.
[8] E. Grädel, Back and forth between logics and games, in: Lectures in Game Theory for Computer Scientists, Springer, 2011, pp. 99–145.
[9] E. Grädel, J. Väänänen, Dependence and Independence, Studia Logica (in press).
[10] L. Henkin, Some remarks on infinitely long formulas, in: Infinitistic Methods, Warsaw, 1961, pp. 167–183.
[11] J. Hintikka, 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) 539–563.
[13] W. Hodges, Logics of imperfect information: why sets of assignments? in: J. van Benthem, B. Löwe, D. Gabbay (Eds.), Interactive Logic, in: Texts in Logic and Games, vol. 1, Amsterdam University Press, 2007, pp. 117–134.
[14] J. Kontinen, Coherence and complexity in fragments of dependence logic, Ph.D. thesis, University of Amsterdam, 2010.
[15] J. Kontinen, J. Väänänen, On definability in dependence logic, Journal of Logic, Language, and Information 18 (2009) 317–332.
[16] J. Kontinen, J. Väänänen, A remark on negation in dependence logic, Notre Dame Journal of Formal Logic 52 (2011) 55–65.
[17] B. Puchala, Infinite two-player games with partial information: logic and algorithms, Diploma thesis, RWTH-Aachen University, http://logic.rwth-aachen.de/~puchala/dipl.pdf, 2008.
[18] J. Reif, The complexity of two-player games of incomplete information, Journal of Computer and System Sciences 29 (1984) 274–301.
[19] J. Väänänen, Dependence Logic, Cambridge University Press, 2007.
[20] W. Walkoe, Finite partially-ordered quantification, Journal of Symbolic Logic 35 (1970) 535–555.