# Logic and Bisimulation for Guarded Teams

Erich Grädel

joint work with Martin Otto

Berkeley, November 2016

# Objective

Bring together two threads of research in logic

- Logics of dependence and independence, based on team semantics
- Guarded logics and bisimulation

# Objective

Bring together two threads of research in logic

- Logics of dependence and independence, based on team semantics
- Guarded logics and bisimulation

We study several notions of guarded teams and guarded team bisimulation.
Our main objective are characterization theorems that relate

- Guarded team logics
- Invariance under guarded team bisimulation
- Fragments of (existential) second-order logic

# Guarded logics

Introduced by Andréka, van Benthem, and Németi (1998)

Guarded logics restrict quantification so that all (sub-)formula only talk about guarded assignments: tuples that are tied together in some atomic fact.

An assignment $s : \{x_1, \ldots, x_k\} \to \mathfrak{A}$ is guarded if there is an atomic fact $\mathfrak{A} \models \beta(a_1, \ldots, a_m)$ such that $\{s(x_1), \ldots, s(x_k)\} \subseteq \{a_1, \ldots, a_m\}$.

# Guarded logics

Introduced by Andréka, van Benthem, and Németi (1998)

Guarded logics restrict quantification so that all (sub-)formula only talk about guarded assignments: tuples that are tied together in some atomic fact.

An assignment $s : \{x_1, \ldots, x_k\} \to \mathfrak{A}$ is guarded if there is an atomic fact $\mathfrak{A} \models \beta(a_1, \ldots, a_m)$ such that $\{s(x_1), \ldots, s(x_k)\} \subseteq \{a_1, \ldots, a_m\}$.

Guarded fragments GF $\subseteq$ FO and $\mu$GF $\subseteq$ LFP:

relativize quantification syntactically to guarded assignments

$\exists y(\mathrm{gd}(\overline{x}, y) \wedge \varphi(\overline{x}, y)) \qquad \forall y(\mathrm{gd}(\overline{x}, y) \to \varphi(\overline{x}, y))$

# Guarded logics

Introduced by Andréka, van Benthem, and Németi (1998)

Guarded logics restrict quantification so that all (sub-)formula only talk about guarded assignments: tuples that are tied together in some atomic fact.

An assignment $s : \{x_1, \ldots, x_k\} \to \mathfrak{A}$ is guarded if there is an atomic fact $\mathfrak{A} \models \beta(a_1, \ldots, a_m)$ such that $\{s(x_1), \ldots, s(x_k)\} \subseteq \{a_1, \ldots, a_m\}$.

Guarded fragments GF $\subseteq$ FO and $\mu$GF $\subseteq$ LFP:

relativize quantification syntactically to guarded assignments

$\exists y(\mathrm{gd}(\overline{x}, y) \wedge \varphi(\overline{x}, y)) \qquad \forall y(\mathrm{gd}(\overline{x}, y) \to \varphi(\overline{x}, y))$

Guarded logics preserve, and to some extent even explain, the good algorithmic and model-theoretic properties of modal logics, such as propositional modal logic ML and the modal $\mu$-calculus $L_\mu$, and lift these to a much more powerful setting.

# Algorithmic and model-theoretic properties of GF and $\mu$GF

- ML $\subseteq$ GF $\subseteq$ FO and $L_\mu \subseteq \mu$GF $\subseteq$ LFP.
- The satisfiability problems for GF and $\mu$GF are decidable.
- GF has the finite model property.
- Tree model property: Every satisfiable formula in GF or $\mu$GF has a model of bounded tree width.
- Automata based decision procedures.
- Efficient model checking via evaluation games of moderate size.

# Algorithmic and model-theoretic properties of GF and $\mu$GF

- ML $\subseteq$ GF $\subseteq$ FO and $L_\mu \subseteq \mu$GF $\subseteq$ LFP.
- The satisfiability problems for GF and $\mu$GF are decidable.
- GF has the finite model property.
- Tree model property: Every satisfiable formula in GF or $\mu$GF has a model of bounded tree width.
- Automata based decision procedures.
- Efficient model checking via evaluation games of moderate size.

Besides GF and $\mu$GF, there also exist extensions to more powerful variants of guarded logics such as loosely guarded and clique guarded logics, and logics with guarded negation.

# Guarded bisimulation

A guarded bisimulation between two structures $\mathfrak{A}$ and $\mathfrak{B}$ is a set of local isomorphisms between guarded assignments, satisfying appropriate back and forth properties.

$\mathfrak{A}, s \sim_g \mathfrak{B}, t$ if there is a guarded bisimulation that relates $s$ and $t$

Further we write $\mathfrak{A} \sim_g \mathfrak{B}$, in the case that $\mathfrak{A}, \varnothing \sim_g \mathfrak{B}, \varnothing$.

# Guarded bisimulation

A guarded bisimulation between two structures $\mathfrak{A}$ and $\mathfrak{B}$ is a set of local isomorphisms between guarded assignments, satisfying appropriate back and forth properties.

$\mathfrak{A}, s \sim_g \mathfrak{B}, t$ if there is a guarded bisimulation that relates $s$ and $t$

Further we write $\mathfrak{A} \sim_g \mathfrak{B}$, in the case that $\mathfrak{A}, \emptyset \sim_g \mathfrak{B}, \emptyset$.

Guarded logics $\mathrm{GF} \subseteq \mathrm{FO}$ and $\mu\mathrm{GF} \subseteq \mathrm{LFP}$ are invariant under guarded bisimulation and satisfy van Benthem style characterization theorems:

$$\mathrm{GF} = \mathrm{FO} \,/\, \sim_g \qquad \text{and} \qquad \mu\mathrm{GF} = \mathrm{GSO} \,/\, \sim_g$$

Can we get similar results in team semantics?

# Team semantics

Invented by Wilfrid Hodges (1997)

Formulae $\psi(x_1, \ldots, x_k)$ are not evaluated on a single assignment $s : \{x_1, \ldots, x_k\} \to A$, but on a set of such assignments, called a team.

$\mathfrak{A} \models_X \varphi(\overline{x})$: $\varphi$ is true in the structure $\mathfrak{A}$ for the team $X$

Motivation: Provide a model-theoretic (compositional) semantics for the independence-friendly logic IF.

Dependency statements such as "$x$ depends on $y$" or that "$x$ and $y$ are independent" do not make much sense for single assignments, but require larger amounts of data, as provided for instance by teams.

Basis of modern logics for dependence and independence (Väänänen 2007)

# Logics of dependence and independence

Formalize dependencies as atomic statements on teams (rather than by annotations of quantifiers):

Dependence:

$$\models_X =(\overline{x}, \overline{y}) \quad :\Longleftrightarrow \quad (\forall s \in X)(\forall s' \in X)(s(\overline{x}) = s'(\overline{x}) \to s(\overline{y}) = s'(\overline{y}))$$

Inclusion:

$$\models_X (\overline{x} \subseteq \overline{y}) \quad :\Longleftrightarrow \quad (\forall s \in X)(\exists s' \in X)(s(\overline{x}) = s'(\overline{y}))$$

Exclusion:

$$\models_X (\overline{x} \mid \overline{y}) \quad :\Longleftrightarrow \quad (\forall s \in X)(\forall s' \in X)(s(\overline{x}) \neq s'(\overline{y}))$$

Independence:

$$\models_X (\overline{x} \perp \overline{y}) \quad :\Longleftrightarrow \quad X(\overline{x}\overline{y}) = X(\overline{x}) \times X(\overline{y})$$

Here $X(\overline{x})$ is the set of all values $s(\overline{x})$ for $s \in X$.

# Logics of dependence and independence

Combine such atoms with logical connectives and quantifiers to obtain full-fledged logics for reasoning about dependence and independence.

Dependence logic: FO + dependence atoms $=(\overline{x}, y)$

Independence logic: FO + independence atoms $\overline{x} \perp \overline{y}$

Inclusion logic: FO + inclusion atoms $(\overline{x} \subseteq \overline{y})$ and so on.

# Logics of dependence and independence

Combine such atoms with logical connectives and quantifiers to obtain full-fledged logics for reasoning about dependence and independence.

Dependence logic: FO + dependence atoms $=(\overline{x}, y)$

Independence logic: FO + independence atoms $\overline{x} \perp \overline{y}$

Inclusion logic: FO + inclusion atoms $(\overline{x} \subseteq \overline{y})$ and so on.

All these logics require team semantics.

What precisely does it mean that, $\mathfrak{A} \models_X \psi(\overline{x})$ ?

# Logics of dependence and independence

Combine such atoms with logical connectives and quantifiers to obtain full-fledged logics for reasoning about dependence and independence.

Dependence logic: FO + dependence atoms $=(\overline{x}, y)$

Independence logic: FO + independence atoms $\overline{x} \perp \overline{y}$

Inclusion logic: FO + inclusion atoms $(\overline{x} \subseteq \overline{y})$ and so on.

All these logics require team semantics.

What precisely does it mean that, $\mathfrak{A} \models_X \psi(\overline{x})$ ? Not in this talk!

# Team semantics

For pure FO (without dependency atoms), team semantics does not produce anything significantly new:

Flatness: $\mathfrak{A} \models_X \psi \iff \mathfrak{A} \models_s \psi(\overline{x})$ for all $s \in X$.

# Team semantics

For pure FO (without dependency atoms), team semantics does not produce anything significantly new:

Flatness: $\mathfrak{A} \models_X \psi \iff \mathfrak{A} \models_s \psi(\overline{x})$ for all $s \in X$.

However, in the presence of non-flat dependencies, team semantics gives second-order features to logics with first-order syntax.

In particular, disjunction splits the team in an arbitrary way:

$$\mathfrak{A} \models_X \psi \vee \varphi \quad :\iff \quad X = Y \cup Z \text{ such that } \mathfrak{A} \models_Y \psi \text{ and } \mathfrak{A} \models_Z \varphi$$

# Team semantics

For pure FO (without dependency atoms), team semantics does not produce anything significantly new:

Flatness: $\mathfrak{A} \models_X \psi \iff \mathfrak{A} \models_s \psi(\overline{x})$ for all $s \in X$.

However, in the presence of non-flat dependencies, team semantics gives second-order features to logics with first-order syntax.

In particular, disjunction splits the team in an arbitrary way:

$$\mathfrak{A} \models_X \psi \vee \varphi \quad :\iff \quad X = Y \cup Z \text{ such that } \mathfrak{A} \models_Y \psi \text{ and } \mathfrak{A} \models_Z \varphi$$

Syntactically extremely simple formulae can express complicated properties:

- 3-SAT is expressibly by $=\!(x, y) \vee =\!(x, y) \vee =\!(u, z)$
  (a disjunction of three dependency atoms).

# Team semantics

For pure FO (without dependency atoms), team semantics does not produce anything significantly new:

Flatness: $\mathfrak{A} \models_X \psi \iff \mathfrak{A} \models_s \psi(\overline{x})$ for all $s \in X$.

However, in the presence of non-flat dependencies, team semantics gives second-order features to logics with first-order syntax.

In particular, disjunction splits the team in an arbitrary way:

$$\mathfrak{A} \models_X \psi \vee \varphi \quad :\iff \quad X = Y \cup Z \text{ such that } \mathfrak{A} \models_Y \psi \text{ and } \mathfrak{A} \models_Z \varphi$$

Syntactically extremely simple formulae can express complicated properties:

- 3-SAT is expressibly by $=(x, y) \vee =(x, y) \vee =(u, z)$
  (a disjunction of three dependency atoms).
- With an inclusion atom, ill-foundedness of a partial order is expressible by
  $\exists x \exists y (y < x \wedge y \subseteq x)$

# From team semantics to Tarski semantics

A team $X$ of assignments $s : V \to A$ can be viewed as a relation $X \subseteq A^{|V|}$.

A formula $\psi(x_1, \ldots, x_n)$ in a logic with team semantics, with vocabulary $\tau$, can be translated into a $\Sigma^1_1$-sentence $\psi^*(X)$ of vocabulary $\tau \cup \{X\}$ such that

$$\mathfrak{A} \models_X \psi(\overline{x}) \quad \Longleftrightarrow \quad (\mathfrak{A}, X) \models \psi^*(X)$$

The expressive power of a logic L with team semantics can be understood by describing the appropriate fragment $F \subseteq \Sigma^1_1$ to which $L$ is equivalent (in the sense just described).

# From team semantics to Tarski semantics

A team $X$ of assignments $s : V \to A$ can be viewed as a relation $X \subseteq A^{|V|}$.

A formula $\psi(x_1, \ldots, x_n)$ in a logic with team semantics, with vocabulary $\tau$, can be translated into a $\Sigma^1_1$-sentence $\psi^*(X)$ of vocabulary $\tau \cup \{X\}$ such that

$$\mathfrak{A} \models_X \psi(\overline{x}) \quad \Longleftrightarrow \quad (\mathfrak{A}, X) \models \psi^*(X)$$

The expressive power of a logic L with team semantics can be understood by describing the appropriate fragment $F \subseteq \Sigma^1_1$ to which $L$ is equivalent (in the sense just described).

Remark. If we add further connectives, such as classical negation and certain variants of implications, we may have to go up to full SO rather than $\Sigma^1_1$

# Team semantics and existential second-order logic

- Exclusion logic and dependence logic capture precisely the downwards closed $\Sigma^1_1$-properties of teams (sentences in which the team predicate appears only negatively). (Kontinen and Väänänen, Galliani)

# Team semantics and existential second-order logic

- Exclusion logic and dependence logic capture precisely the downwards closed $\Sigma^1_1$-properties of teams (sentences in which the team predicate appears only negatively). (Kontinen and Väänänen, Galliani)
- FO + inclusion + exclusion $\equiv$ independence logic $\equiv$ $\Sigma^1_1$.
  Thus, all NP-properties of teams are definable in independence logic.
  (Grädel and Väänänen, Galliani)

# Team semantics and existential second-order logic

- Exclusion logic and dependence logic capture precisely the downwards closed $\Sigma^1_1$-properties of teams (sentences in which the team predicate appears only negatively). (Kontinen and Väänänen, Galliani)
- FO + inclusion + exclusion $\equiv$ independence logic $\equiv$ $\Sigma^1_1$.
  Thus, all NP-properties of teams are definable in independence logic.
  (Grädel and Väänänen, Galliani)
- Inclusion logic captures the posGFP-fragment of LFP:
  Thus, on finite structures, inclusion logic coincides with LFP
  (Galliani and Hella)

# Team semantics and existential second-order logic

- Exclusion logic and dependence logic capture precisely the downwards closed $\Sigma^1_1$-properties of teams (sentences in which the team predicate appears only negatively). (Kontinen and Väänänen, Galliani)
- FO + inclusion + exclusion $\equiv$ independence logic $\equiv$ $\Sigma^1_1$.
  Thus, all NP-properties of teams are definable in independence logic.
  (Grädel and Väänänen, Galliani)
- Inclusion logic captures the posGFP-fragment of LFP:
  Thus, on finite structures, inclusion logic coincides with LFP
  (Galliani and Hella)
- On finite structures, inclusion logic with counting coincides with fixed-point logic with counting: $\mathrm{FO}(\subseteq, \exists^{\geq\mu}) \equiv \mathrm{FPC}$
  (Grädel and Hegselmann)

# Guarded FO for teams

Guarded teams: contain only guarded assignments $s : \{x_1, \ldots, x_k\} \to A$, which means that all values $s(x_i)$ co-exist in some atomic fact of $\mathfrak{A}$.

Guarded team semantics for FO: $\mathfrak{A} \models_X^{\mathrm{hg}} \varphi(\overline{x})$.
Restrict semantic rules to guarded teams

Guarded fragment GF: relativize quantification syntactically to guarded assignments: $\exists y(\mathrm{gd}(\overline{x}, y) \wedge \varphi(\overline{x}, y))$ $\forall y(\mathrm{gd}(\overline{x}, y) \to \varphi(\overline{x}, y))$

Proposition. Every $\varphi(\overline{x}) \in \mathrm{FO}$ can be translated into $\varphi^{\mathrm{hg}}(\overline{x}) \in \mathrm{GF}$ such that, for all guarded teams $X$, $\mathfrak{A} \models_X^{\mathrm{hg}} \varphi(\overline{x}) \iff \mathfrak{A} \models_X \varphi^{\mathrm{hg}}(\overline{x})$

# Two notions of guarded team bisimulation

$\mathfrak{A}, X \sim_g \mathfrak{B}, Y$: $X$, and $Y$ contain precisely the same guarded bisimulation types, i.e., for all $s \in X$ there exists $t \in Y$, and vice versa, such that $\mathfrak{A}, s \sim_g \mathfrak{B}, t$.

# Two notions of guarded team bisimulation

$\mathfrak{A}, X \sim_g \mathfrak{B}, Y$: $X$, and $Y$ contain precisely the same guarded bisimulation types, i.e., for all $s \in X$ there exists $t \in Y$, and vice versa, such that $\mathfrak{A}, s \sim_g \mathfrak{B}, t$.

$\mathfrak{A}, X \approx_g \mathfrak{B}, Y$: The expanded structures $(\mathfrak{A}, X)$ and $(\mathfrak{B}, Y)$ are $\sim_g$-bisimilar (in the traditional sense, but taking into account the predicate for the team)

# Two notions of guarded team bisimulation

$\mathfrak{A}, X \sim_g \mathfrak{B}, Y$: $X$, and $Y$ contain precisely the same guarded bisimulation types, i.e., for all $s \in X$ there exists $t \in Y$, and vice versa, such that $\mathfrak{A}, s \sim_g \mathfrak{B}, t$.

$\mathfrak{A}, X \approx_g \mathfrak{B}, Y$: The expanded structures $(\mathfrak{A}, X)$ and $(\mathfrak{B}, Y)$ are $\sim_g$-bisimilar (in the traditional sense, but taking into account the predicate for the team)

Obviously, $\mathfrak{A}, X \approx_g \mathfrak{B}, Y \implies \mathfrak{A}, X \sim_g \mathfrak{B}, Y$.
Thus, if $\varphi(\overline{x})$ is $\sim_g$-invariant, then it is also $\approx_g$-invariant

Proposition. The converse ist not true.
In fact inclusions $\overline{x} \subseteq \overline{y}$ and exclusions $\overline{x} \mid \overline{y}$ are $\approx_g$-invariant, but not $\sim_g$-invariant.

# Comparing team bisimulations

$\mathfrak{A}$ :

$$a \longrightarrow b \longrightarrow c$$

$\mathfrak{B}$ :

$$\begin{tikzcd} a' \arrow[r] & b' \arrow[dr] & \\ & & c' \\ a'' \arrow[r] & b'' \arrow[ur] & \end{tikzcd}$$

# Comparing team bisimulations

$\mathfrak{A}$ :

$$a \longrightarrow b \longrightarrow c$$

$\mathfrak{B}$ :

$$\begin{tikzcd} a' \arrow[r] & b' \arrow[dr] & \\ & & c' \\ a'' \arrow[r] & b'' \arrow[ur] & \end{tikzcd}$$

Bisimilar assignments: $\mathfrak{A}, (ab) \sim_g \mathfrak{B}, (a'b')$

# Comparing team bisimulations

$\mathfrak{A}$ :

$$a \longrightarrow b \longrightarrow c$$

$\mathfrak{B}$ :

$$\begin{tikzcd} a' \arrow[r] & b' \arrow[dr] & \\ & & c' \\ a'' \arrow[r] & b'' \arrow[ur] & \end{tikzcd}$$

Bisimilar assignments: $\mathfrak{A}, (ab) \sim_g \mathfrak{B}, (a'b')$ and also $\mathfrak{A}, (bc) \sim_g \mathfrak{B}, (b''c')$

# Comparing team bisimulations

$\mathfrak{A}$ :

$$a \longrightarrow b \longrightarrow c$$

$\mathfrak{B}$ :

$$\begin{tikzcd} a' \arrow[r] & b' \arrow[dr] & \\ & & c' \\ a'' \arrow[r] & b'' \arrow[ur] & \end{tikzcd}$$

Bisimilar assignments: $\mathfrak{A}, (ab) \sim_g \mathfrak{B}, (a'b')$ and also $\mathfrak{A}, (bc) \sim_g \mathfrak{B}, (b''c')$

For the teams $X = \{(ab), (bc)\}$ and $Y = \{(a'b'), (b''c')\}$ we thus have $\mathfrak{A}, X \sim_g \mathfrak{B}, Y$.

# Comparing team bisimulations

$\mathfrak{A}$ :

$$a \longrightarrow b \longrightarrow c$$

$\mathfrak{B}$ :

$$\begin{tikzcd} a' \arrow[r] & b' \arrow[dr] & \\ & & c' \\ a'' \arrow[r] & b'' \arrow[ur] & \end{tikzcd}$$

Bisimilar assignments: $\mathfrak{A}, (ab) \sim_g \mathfrak{B}, (a'b')$ and also $\mathfrak{A}, (bc) \sim_g \mathfrak{B}, (b''c')$

For the teams $X = \{(ab), (bc)\}$ and $Y = \{(a'b'), (b''c')\}$ we thus have $\mathfrak{A}, X \sim_g \mathfrak{B}, Y$. But clearly not that $(\mathfrak{A}, X) \approx_g (\mathfrak{B}, Y)$.

# Comparing team bisimulations

$\mathfrak{A}$ :

$$a \longrightarrow b \longrightarrow c$$

$\mathfrak{B}$ :

$$\begin{tikzcd} a' \arrow[r] & b' \arrow[dr] & \\ & & c' \\ a'' \arrow[r] & b'' \arrow[ur] & \end{tikzcd}$$

Bisimilar assignments: $\mathfrak{A}, (ab) \sim_g \mathfrak{B}, (a'b')$ and also $\mathfrak{A}, (bc) \sim_g \mathfrak{B}, (b''c')$

For the teams $X = \{(ab), (bc)\}$ and $Y = \{(a'b'), (b''c')\}$ we thus have $\mathfrak{A}, X \sim_g \mathfrak{B}, Y$. But clearly not that $(\mathfrak{A}, X) \approx_g (\mathfrak{B}, Y)$.

Notice that $\mathfrak{B} \models_Y (x \mid y)$, but $\mathfrak{A} \not\models_X (x \mid y)$.
Hence exclusion atoms are not $\sim_g$-invariant.

An almost identical argument shows the same for inclusion atoms.

# Invariance under strong team bisimulation

Inclusion and exclusion dependencies are $\mathrm{GF}^X$-definable, and $\approx_g$-invariant.

For a team $X$ with variables $x, y, \bar{z}$ we have that

$$\mathfrak{A} \models_X (x \subseteq y) \iff (\mathfrak{A}, X) \models \forall x y \bar{z} (X x y \bar{z} \to \exists u \bar{v} X u x \bar{v})$$

$$\mathfrak{A} \models_X (x \mid y) \iff (\mathfrak{A}, X) \models \forall x y \bar{z} (X x y \bar{z} \to \neg \exists u \bar{v} X u x \bar{v})$$

# Invariance under strong team bisimulation

Inclusion and exclusion dependencies are $\mathrm{GF}^X$-definable, and $\approx_g$-invariant.

For a team $X$ with variables $x, y, \bar{z}$ we have that

$$\mathfrak{A} \models_X (x \subseteq y) \iff (\mathfrak{A}, X) \models \forall x y \bar{z} (X x y \bar{z} \to \exists u \bar{v} X u x \bar{v})$$
$$\mathfrak{A} \models_X (x \mid y) \iff (\mathfrak{A}, X) \models \forall x y \bar{z} (X x y \bar{z} \to \neg \exists u \bar{v} X u x \bar{v})$$

However, $\approx_g$ is, in general, not compatible with team disjunction.

Proposition. $(x \mid y) \vee (x \mid y)$ is not $\approx_g$-invariant.

# Invariance under strong team bisimulation

Inclusion and exclusion dependencies are $\mathrm{GF}^X$-definable, and $\approx_g$-invariant.

For a team $X$ with variables $x, y, \bar{z}$ we have that

$$\mathfrak{A} \models_X (x \subseteq y) \iff (\mathfrak{A}, X) \models \forall x y \bar{z} (X x y \bar{z} \to \exists u \bar{v} X u x \bar{v})$$
$$\mathfrak{A} \models_X (x \mid y) \iff (\mathfrak{A}, X) \models \forall x y \bar{z} (X x y \bar{z} \to \neg \exists u \bar{v} X u x \bar{v})$$

However, $\approx_g$ is, in general, not compatible with team disjunction.

Proposition. $(x \mid y) \vee (x \mid y)$ is not $\approx_g$-invariant.

Let $C_n$ be the directed cycle of length $n$, and $X_n$ the team of all its edges.

- $(C_m, X_m) \approx_g (C_n, X_n)$ for all $m, n > 2$, but
- $C_n \models_{X_n} (x \mid y) \vee (x \mid y) \iff n$ is even

# Characterization Theorems

We aim at results that combine characterizations of bisimulation invariance (van Benthem style) with translations between team semantics and Tarski semantics.

# Characterization Theorems

We aim at results that combine characterizations of bisimulation invariance (van Benthem style) with translations between team semantics and Tarski semantics.

Theorem. $\mathrm{FO}/\sim_g \equiv \mathrm{FO}^{\mathrm{hg}} \equiv \mathrm{GF} \equiv [\mathrm{GF}]^X \equiv [\mathrm{FO}]^X/\sim_g$.

Here $[\mathrm{GF}]^X$ is the set of all sentences $\forall\overline{x}(X\overline{x} \to \varphi(\overline{x}))$ (with Tarski semantics) such that $\varphi(\overline{x})$ in GF.

All these formalisms describe the same properties of horizontally guarded teams. These properties are downwards closed and $\sim_g$-invariant.

# Characterization Theorems

We aim at results that combine characterizations of bisimulation invariance (van Benthem style) with translations between team semantics and Tarski semantics.

Theorem. $\mathrm{FO}/\sim_g \equiv \mathrm{FO}^{\mathrm{hg}} \equiv \mathrm{GF} \equiv [\mathrm{GF}]^X \equiv [\mathrm{FO}]^X/\sim_g$.

Here $[\mathrm{GF}]^X$ is the set of all sentences $\forall\overline{x}(X\overline{x} \to \varphi(\overline{x}))$ (with Tarski semantics) such that $\varphi(\overline{x})$ in GF.

All these formalisms describe the same properties of horizontally guarded teams. These properties are downwards closed and $\sim_g$-invariant.

Proofs make use of Martin Otto's general method for proving characterization theorems, showing that a $\sim_g$-equivalent FO-formula is in fact already $\sim_g^\ell$-invariant, for some fixed $\ell \in \mathbb{N}$. The characterization theorems hold in the general as well as in the finite model theory setting.

# A second characterization theorem

A stronger case concerns the $\sim_g$-invariant properties of guarded teams that are definable by sentences $\psi(X) \in \mathrm{FO}^X$ (with Tarski semantics).

These are not necessarily downwards closed. We need to expand FO (with team semantics) by classical negation: $\mathfrak{A} \models_X \mathrm{non}(\varphi) \iff \mathfrak{A} \not\models_X \varphi$

Theorem. $\mathrm{FO}_{\mathrm{non}}/\sim_g \equiv \mathrm{FO}^{\mathrm{hg}}_{\mathrm{non}} \equiv \mathrm{GF}_{\mathrm{non}} \equiv \mathrm{FO}^X/\sim_g$.

# A second characterization theorem

A stronger case concerns the $\sim_g$-invariant properties of guarded teams that are definable by sentences $\psi(X) \in \mathrm{FO}^X$ (with Tarski semantics).

These are not necessarily downwards closed. We need to expand FO (with team semantics) by classical negation: $\mathfrak{A} \models_X \mathrm{non}(\varphi) \iff \mathfrak{A} \not\models_X \varphi$

Theorem. $\mathrm{FO}_{\mathrm{non}}/\sim_g \equiv \mathrm{FO}^{\mathrm{hg}}_{\mathrm{non}} \equiv \mathrm{GF}_{\mathrm{non}} \equiv \mathrm{FO}^X/\sim_g$.

It is tempting to assume that this is also equivalent to $\mathrm{GF}^X$, i.e., to sentences $\psi(X) \in \mathrm{GF}$ (with Tarski semantics).

But no! $\mathrm{GF}^X$ is only $\approx_g$-invariant, and not $\sim_g$-invariant.

Instead we can use the fragment of $\mathrm{GF}^X$, consisting of the Hintikka-sentences saying which $\sim_g$-types occur in $X$.

# Characterization theorems for $\approx_g$-invariance?

Proposition. $\mathrm{FO}/\approx_g \equiv \mathrm{FO}/\sim_g$

Indeed every $\approx_g$-invariant property that is flat, is in fact also $\sim_g$-invariant.

# Characterization theorems for $\approx_g$-invariance?

Proposition. $\mathrm{FO}/\approx_g \equiv \mathrm{FO}/\sim_g$

Indeed every $\approx_g$-invariant property that is flat, is in fact also $\sim_g$-invariant.

Guarded fixed-point logic $\mu\mathrm{GF}^X$ is also $\approx_g$-invariant and satisfies a characterization theorem: $\mu\mathrm{GF}^X \equiv \mathrm{GSO}^X/\approx_g$

Can we obtain a characterization theorem relating guarded inclusion logic with (a fragment of) guarded fixed-point logic?

Proposition. $\mathrm{FO}(\subseteq)^{\mathrm{hg}} \equiv \mathrm{GF}(\subseteq) \leq \mu\mathrm{GF}^X$.

Formulae $\varphi(\overline{x}) \in \mathrm{GF}(\subseteq)$ are translated into sentences $\forall\overline{x}(X\overline{x} \to \psi(X,\overline{x}))$ where $\psi(X,\overline{x})$ has only greatest fixed-points and is positive in $X$.

Questions: Is this fragment equivalent with $\mathrm{GF}(\subseteq)$ ? And with $\mathrm{FO}(\subseteq)/\approx_g$ ?

# Outlook

Vertically guarded teams: For every variable $x$, the set $X(x)$ is guarded, where guarded sets are given by an arbitrary hypergraph $(A, V)$ (that can a priori be completely independent of the relations of $\mathfrak{A}$)

Vertically guarded team semantics: $\mathfrak{A} \models^{vg}_{X} \varphi(\overline{x})$
modify semantic rules such that all teams remain vertically guarded

Question: Expressive power of $\mathrm{FO}^{vg}$ ?

Strongly depends on the hypergraph defining the guarded sets.

In the extreme cases (all sets are guarded or only singletons are guarded), $\mathrm{FO}^{vg}$ collapses to FO.

But we also know cases where $\mathrm{FO}^{vg}$ is extremely powerful, up to (almost) full second-order logic.