Strongly First-Order Dependencies in Team Semantics

Pietro Galliani

TU Clausthal

Dependence Logic Academy Colloquium
A common conversation

A Dependece Logic is First-Order Logic plus functional dependence atoms.

B OK.

A And it is as expressive as the existential fragment of Second-Order Logic.

B Why? Functional dependency is a first-order definable property...
A Small Problem

A common conversation

A Dependence Logic is First-Order Logic plus functional dependence atoms.

B OK.

A And it is as expressive as the existential fragment of Second-Order Logic.

B Why? Functional dependency is a first-order definable property...
Dependence Logic is First-Order Logic plus functional dependence atoms.

OK.

And it is as expressive as the existential fragment of Second-Order Logic.

Why? Functional dependency is a first-order definable property...
A Dependence Logic is First-Order Logic plus functional dependence atoms.

B OK.

A And it is as expressive as the existential fragment of Second-Order Logic.

B Why? Functional dependency is a first-order definable property...
Actually…

Dependence Logic is First-Order Logic with team semantics plus functional dependence atoms.

Team Semantics

- Extends Tarski’s semantics to sets of assignments;
- Second-order character (see rules for disjunctions and existentials);
- Equivalent to Tarski’s semantics wrt First Order sentences, but allows for different extensions.
Actually…

Dependence Logic is First-Order Logic *with team semantics* plus functional dependence atoms.

Team Semantics

- Extends Tarski’s semantics to *sets* of assignments;
- Second-order character (see rules for disjunctions and existentials);
- Equivalent to Tarski’s semantics wrt First Order *sentences*, but allows for different extensions.
What this talk is about

- **Team Semantics**: extends Tarski, satisfaction wrt sets of assignments: $X = \{s_1 \ldots s_n\}$, $M \models_X \phi$ iff . . .

- Adding *Dependency atoms*: $\mathcal{D} = \{D_1, \ldots, D_n\}$, $FO(\mathcal{D})$ can be much stronger than $FO$ (often, $FO(\mathcal{D}) \equiv \Sigma^1_1$), even if the $D_i$ *themselves are first-order* (as properties of relations);

- **Main question**: find nontrivial $\mathcal{D}$ s.t. $FO(\mathcal{D})$ captures only first order properties of relations (and, therefore, $FO(\mathcal{D}) = FO$ wrt sentences).
Team Semantics: extends Tarski, satisfaction wrt sets of assignments: $X = \{s_1 \ldots s_n\}$, $M \models_X \phi$ iff . . .

Adding Dependency atoms: $\mathcal{D} = \{D_1, \ldots, D_n\}$, $FO(\mathcal{D})$ can be much stronger than $FO$ (often, $FO(\mathcal{D}) \equiv \Sigma^1_1$), even if the $D_i$ themselves are first-order (as properties of relations);

Main question: find nontrivial $\mathcal{D}$ s.t. $FO(\mathcal{D})$ captures only first order properties of relations (and, therefore, $FO(\mathcal{D}) = FO$ wrt sentences).
• **Team Semantics**: extends Tarski, satisfaction wrt *sets* of assignments: $X = \{s_1 \ldots s_n\}$, $M \models_X \phi$ iff . . .

• Adding **Dependency atoms**: $\mathcal{D} = \{D_1, \ldots, D_n\}$, $FO(\mathcal{D})$ can be much stronger than $FO$ (often, $FO(\mathcal{D}) \equiv \Sigma^1_1$), *even if the* $D_i$ *themselves are first-order* (as properties of relations);

• **Main question**: find nontrivial $\mathcal{D}$ s.t. $FO(\mathcal{D})$ captures only first order properties of relations (and, therefore, $FO(\mathcal{D}) = FO$ wrt sentences).
Why?

Answer One: Applications
People have been thinking about applications of team semantics to database theory, belief representation, and even physics (more about this later in this conference). It would be useful to know which dependencies are “safe”.

Answer Two: the First Order / Second Order Barrier
Studying Team Semantics-based extensions of First Order Logic gives us a new perspective about the boundary between first and second order; and in this talk, we will attempt to approach this boundary “from below”.

Pietro Galliani
Strongly First-Order Dependencies in Team Semantics
Why?

Answer One: Applications

People have been thinking about applications of team semantics to database theory, belief representation, and even physics (more about this later in this conference). It would be useful to know which dependencies are “safe”.

Answer Two: the First Order / Second Order Barrier

Studying Team Semantics-based extensions of First Order Logic gives us a new perspective about the boundary between first and second order; and in this talk, we will attempt to approach this boundary “from below”.
1 Team Semantics and Dependencies

2 Upwards Closed Dependencies

3 Boundedness
1. Team Semantics and Dependencies
2. Upwards Closed Dependencies
3. Boundedness
Tarski’s Semantics for First Order Logic

Assignments as States of Things

An assignment $s$ is a function from variables to elements of a model, representing a possible state of things. $M \models_s \phi$ if and only if $\phi$ is true of the state $s$.

Tarski Semantics (assuming Negation Normal Form)

- $M \models_s R \vec{t}$ if and only if $\vec{t}(s) \in R^M$;
- $M \models_s \neg R \vec{t}$ if and only if $\vec{t}(s) \notin R^M$;
- $M \models_s t_1 = t_2$ if and only if $t_1(s) = t_2(s)$;
- $M \models_s t_1 \neq t_2$ if and only if $t_1(s) = t_2(s)$;
- $M \models_s \phi \land \psi$ if and only if $M \models_s \phi$ and $M \models_s \psi$;
- $M \models_s \phi \lor \psi$ if and only if $M \models_s \phi$ or $M \models_s \psi$;
- $M \models_s \exists v \phi$ if and only if $\exists m \in \text{Dom}(M)$ s.t. $M \models_{s[m/v]} \phi$;
- $M \models_s \forall v \phi$ if and only if $\forall m \in \text{Dom}(M), M \models_{s[m/v]} \phi$. 
Tarski’s Semantics for First Order Logic

Assignments as States of Things

An assignment $s$ is a function from variables to elements of a model, representing a possible state of things. $M \models_s \phi$ if and only if $\phi$ is true of the state $s$.

Tarski Semantics (assuming Negation Normal Form)

- $M \models_s R\vec{t}$ if and only if $\vec{t}(s) \in R^M$;
- $M \models_s \neg R\vec{t}$ if and only if $\vec{t}(s) \notin R^M$;
- $M \models_s t_1 = t_2$ if and only if $t_1(s) = t_2(s)$;
- $M \models_s t_1 \neq t_2$ if and only if $t_1(s) = t_2(s)$;
- $M \models_s \phi \land \psi$ if and only if $M \models_s \phi$ and $M \models_s \psi$;
- $M \models_s \phi \lor \psi$ if and only if $M \models_s \phi$ or $M \models_s \psi$;
- $M \models_s \exists v \phi$ if and only if $\exists m \in \text{Dom}(M)$ s.t. $M \models_s[m/v] \phi$;
- $M \models_s \forall v \phi$ if and only if $\forall m \in \text{Dom}(M), M \models_s[m/v] \phi$. 

Pietro Galliani

Strongly First-Order Dependencies in Team Semantics
From Assignments to Teams

Teams (= Belief Sets)
A team $X$ is a set of assignments (possible states of things). It may be thought of as the belief state of an agent.

Satisfaction in a Team
For all first order $\phi$, $M \models_X \phi$ if and only if $M \models_s \phi$ for all $s \in X$.

Operations over Teams
- Supplementation: $X[H/v] = \{s[m/v] : s \in X, m \in H(s)\}$;
- Duplication: $X[M/v] = \{s[m/v] : s \in X, m \in \text{Dom}(M)\}$.

From Teams to Relations
For $X$ team, $\vec{v}$ variables, $X(\vec{v}) = \{s(\vec{v}) : s \in X\}$.
From Assignments to Teams

**Teams (= Belief Sets)**

A team $X$ is a set of assignments (possible states of things). It may be thought of as the belief state of an agent.

**Satisfaction in a Team**

For all first order $\phi$, $M \models_X \phi$ if and only if $M \models_s \phi$ for all $s \in X$.

**Operations over Teams**

- **Supplementation**: $X[H/v] = \{s[m/v] : s \in X, m \in H(s)\}$;
- **Duplication**: $X[M/v] = \{s[m/v] : s \in X, m \in \text{Dom}(M)\}$.

**From Teams to Relations**

For $X$ team, $\vec{v}$ variables, $X(\vec{v}) = \{s(\vec{v}) : s \in X\}$. 
Teams (= Belief Sets)

A team $X$ is a set of assignments (possible states of things). It may be thought of as the belief state of an agent.

Satisfaction in a Team

For all first order $\phi$, $M \models^X \phi$ if and only if $M \models^s \phi$ for all $s \in X$.

Operations over Teams

- **Supplementation:** $X[H/v] = \{s[m/v] : s \in X, m \in H(s)\}$;
- **Duplication:** $X[M/v] = \{s[m/v] : s \in X, m \in \text{Dom}(M)\}$.

From Teams to Relations

For $X$ team, $\vec{v}$ variables, $X(\vec{v}) = \{s(\vec{v}) : s \in X\}$. 

Pietro Galliani

Strongly First-Order Dependencies in Team Semantics
Team Semantics

$M \models^X \phi$ if and only if $M \models^s \phi$ for all $s \in X$.

- If $\alpha$ literal, $M \models^X \alpha$ iff for all $s \in X$, $M \models^s \alpha$;
- $M \models^X \phi \land \psi$ iff for all $s \in X$, $M \models^s \phi$ and $M \models^s \psi$;
- $M \models^X \phi \lor \psi$ iff for all $s \in X$, $M \models^s \phi$ or $M \models^s \psi$;
- $M \models^X \exists v \phi$ iff for all $s \in X$, $\exists m \in \text{Dom}(M)$ s.t. $M \models^s[m/v] \phi$;
- $M \models^X \forall v \phi$ iff for all $s \in X$, $\forall m \in \text{Dom}(M)$, $M \models^s[m/v] \phi$.

Aside: Team Semantics and Game Theoretic Semantics

Teams correspond to sets of possible states in the subgames of the semantic game, when players are allowed nondeterministic strategies.
Team Semantics for First Order Logic

**Team Semantics**

$M \models_X \phi$ if and only if $M \models_s \phi$ for all $s \in X$.

- If $\alpha$ literal, $M \models_X \alpha$ iff for all $s \in X$, $M \models_s \alpha$;
- $M \models_X \phi \land \psi$ iff for all $s \in X$, $M \models_s \phi$ and $M \models_s \psi$;
- $M \models_X \phi \lor \psi$ iff for all $s \in X$, $M \models_s \phi$ or $M \models_s \psi$;
- $M \models_X \exists v \phi$ iff for all $s \in X$, $\exists m \in \text{Dom}(M)$ s.t. $M \models_s[m/v] \phi$;
- $M \models_X \forall v \phi$ iff for all $s \in X$, $\forall m \in \text{Dom}(M)$, $M \models_s[m/v] \phi$.

**Aside: Team Semantics and Game Theoretic Semantics**

Teams correspond to *sets of possible states* in the subgames of the semantic game, when players are allowed nondeterministic strategies.
Team Semantics for First Order Logic

Team Semantics

\[ M \models_X \phi \text{ if and only if } M \models_s \phi \text{ for all } s \in X. \]

- If \( \alpha \) literal, \( M \models_X \alpha \) iff for all \( s \in X \), \( M \models_s \alpha \);
- \( M \models_X \phi \land \psi \) iff \( M \models_X \phi \) and \( M \models_X \psi \);
- \( M \models_X \phi \lor \psi \) iff for all \( s \in X \), \( M \models_s \phi \) or \( M \models_s \psi \);
- \( M \models_X \exists v \phi \) iff for all \( s \in X \), \( \exists m \in \text{Dom}(M) \) s.t. \( M \models_s[m/v] \phi \);
- \( M \models_X \forall v \phi \) iff for all \( s \in X \), \( \forall m \in \text{Dom}(M) \), \( M \models_s[m/v] \phi \).

Aside: Team Semantics and Game Theoretic Semantics

Teams correspond to *sets of possible states* in the subgames of the semantic game, when players are allowed nondeterministic strategies.
Team Semantics for First Order Logic

**Team Semantics**

\[ M \models_X \phi \text{ if and only if } M \models_s \phi \text{ for all } s \in X. \]

- If \( \alpha \) literal, \( M \models_X \alpha \) iff for all \( s \in X \), \( M \models_s \alpha \);
- \( M \models_X \phi \land \psi \) iff \( M \models_X \phi \) and \( M \models_X \psi \);
- \( M \models_X \phi \lor \psi \) iff \( X = Y \cup Z \), \( M \models_Y \phi \) and \( M \models_Z \psi \);
- \( M \models_X \exists v \phi \) iff for all \( s \in X \), \( \exists m \in \text{Dom}(M) \) s.t. \( M \models_s[m/v] \phi \);
- \( M \models_X \forall v \phi \) iff for all \( s \in X \), \( \forall m \in \text{Dom}(M) \), \( M \models_s[m/v] \phi \).

**Aside: Team Semantics and Game Theoretic Semantics**

Teams correspond to *sets of possible states* in the subgames of the semantic game, when players are allowed nondeterministic strategies.
Team Semantics for First Order Logic

Team Semantics

\[ M \models_X \phi \text{ if and only if } M \models_s \phi \text{ for all } s \in X. \]

- If \( \alpha \) literal, \( M \models_X \alpha \) iff for all \( s \in X \), \( M \models_s \alpha \);
- \( M \models_X \phi \land \psi \) iff \( M \models_X \phi \) and \( M \models_X \psi \);
- \( M \models_X \phi \lor \psi \) iff \( X = Y \cup Z \), \( M \models_Y \phi \) and \( M \models_Z \psi \);
- \( M \models_X \exists v \phi \) iff \( \exists H : X \rightarrow \mathcal{P}(\text{Dom}(M)) \setminus \{\emptyset\} \) s.t. \( M \models_X [H/v] \phi \);
- \( M \models_X \forall v \phi \) iff for all \( s \in X \), \( \forall m \in \text{Dom}(M) \), \( M \models_s [m/v] \phi \).

Aside: Team Semantics and Game Theoretic Semantics

Teams correspond to *sets of possible states* in the subgames of the semantic game, when players are allowed nondeterministic strategies.
Team Semantics

\[ M \models_X \phi \text{ if and only if } M \models_s \phi \text{ for all } s \in X. \]

- If \( \alpha \) literal, \( M \models_X \alpha \) iff for all \( s \in X, M \models_s \alpha \);
- \( M \models_X \phi \land \psi \) iff \( M \models_X \phi \) and \( M \models_X \psi \);
- \( M \models_X \phi \lor \psi \) iff \( X = Y \cup Z, M \models_Y \phi \) and \( M \models_Z \psi \);
- \( M \models_X \exists v \phi \) iff \( \exists H : X \rightarrow \mathcal{P}(\text{Dom}(M)) \setminus \{\emptyset\} \) s.t. \( M \models_X[H/v] \phi \);
- \( M \models_X \forall v \phi \) iff \( M \models_X[M/v] \phi \).

Aside: Team Semantics and Game Theoretic Semantics

Teams correspond to sets of possible states in the subgames of the semantic game, when players are allowed nondeterministic strategies.
Team Semantics

\[ M \models_X \phi \text{ if and only if } M \models_s \phi \text{ for all } s \in X. \]

- If \( \alpha \) literal, \( M \models_X \alpha \) iff for all \( s \in X \), \( M \models_s \alpha \);
- \( M \models_X \phi \land \psi \) iff \( M \models_X \phi \) and \( M \models_X \psi \);
- \( M \models_X \phi \lor \psi \) iff \( X = Y \cup Z \), \( M \models_Y \phi \) and \( M \models_Z \psi \);
- \( M \models_X \exists v \phi \) iff \( \exists H : X \to \mathcal{P}({\text{Dom}}(M)) \setminus \{\emptyset\} \) s.t. \( M \models_X[H/v] \phi \);
- \( M \models_X \forall v \phi \) iff \( M \models_X[M/v] \phi \).

Aside: Team Semantics and Game Theoretic Semantics

Teams correspond to \textit{sets of possible states} in the subgames of the semantic game, when players are allowed nondeterministic strategies.
### Satisfaction Conditions in Team Semantics

\[ \| \phi(\vec{v}) \| = \{ (M, X(\vec{v})) : M \vDash_X \phi(\vec{v}) \} \]

### Satisfaction Conditions of F.O. formulas are F.O. Definable

\[ \| \phi(\vec{v}) \| = \{ (M, R) : (M, R) \vDash \forall \vec{v} (R\vec{v} \rightarrow \phi(\vec{v})) \} \]

### However...

Not all first order properties of relations correspond to satisfaction conditions of first order formulas in Team Semantics. For example, \( M \vDash \emptyset \phi \) for all \( \phi \), and thus

\[ \| \phi(\vec{v}) \| \neq \{ (M, R) : R \neq \emptyset \} = \{ (M, R) : (M, R) \vDash \exists \vec{v} R\vec{v} \} \].
Satisfaction Conditions of First Order Formulas

Satisfaction Conditions in Team Semantics

\[ \| \phi(\vec{v}) \| = \{ (M, X(\vec{v})) : M \models_X \phi(\vec{v}) \} \]

Satisfaction Conditions of F.O. formulas are F.O. Definable

\[ \| \phi(\vec{v}) \| = \{ (M, R) : (M, R) \models \forall \vec{v} (R\vec{v} \rightarrow \phi(\vec{v})) \} \].

However...

Not all first order properties of relations correspond to satisfaction conditions of first order formulas in Team Semantics. For example, \( M \models_\emptyset \phi \) for all \( \phi \), and thus

\[ \| \phi(\vec{v}) \| \neq \{ (M, R) : R \neq \emptyset \} = \{ (M, R) : (M, R) \models \exists \vec{v} R\vec{v} \} \].
Satisfaction Conditions in Team Semantics

\[ \| \phi(\vec{v}) \| = \{(M, X(\vec{v})) : M \models_X \phi(\vec{v}) \} \]

Satisfaction Conditions of F.O. formulas are F.O. Definable

\[ \| \phi(\vec{v}) \| = \{(M, R) : (M, R) \models \forall \vec{v} (R\vec{v} \rightarrow \phi(\vec{v})) \}. \]

However...

Not all first order properties of relations correspond to satisfaction conditions of first order formulas in Team Semantics. For example, \( M \models \emptyset \phi \) for all \( \phi \), and thus

\[ \| \phi(\vec{v}) \| \neq \{(M, R) : R \neq \emptyset \} = \{(M, R) : (M, R) \models \exists \vec{v} R\vec{v} \}. \]
What we just saw

There is no first order $\phi$ such that $M \models X \phi$ if and only if $X \neq \emptyset$.

A new atom

$M \models_X \text{NE}$ if and only if $X \neq \emptyset$.

Question

What kind of logic is $\text{FO(NE)}$? Is any sentence of $\text{FO(NE)}$ equivalent to some first order sentence? What about other atoms, for example from Database Theory?
Dependence Logic $\text{FO}(\models(\cdot, \cdot))$: $M \models_X (\vec{y}, \vec{z})$ iff for all $s, s' \in X$, $s(\vec{y}) = s'(\vec{y}) \Rightarrow s(\vec{z}) = s'(\vec{z})$;

Exclusion Logic $\text{FO}(\mid)$: $M \models_X \vec{y} \mid \vec{z}$ iff $X(\vec{y}) \cap X(\vec{z}) = \emptyset$;

Inclusion Logic $\text{FO}(\subseteq)$: $M \models_X \vec{y} \subseteq \vec{z}$ iff $X(\vec{y}) \subseteq X(\vec{z})$;

Independence Logic $\text{FO}(\perp)$: $M \models_X \vec{y} \perp \vec{z}$ iff $X(\vec{y}\vec{z}) = X(\vec{y}) \times X(\vec{z})$.

First Order?

These dependencies are first-order (as properties of relations), but the resulting logics are not: for example, Independence Logic $= \Sigma^1_1$ and Inclusion Logic $= GFP^+$.

The Question

Under which conditions is $\text{FO}(\mathcal{D})$ equivalent to $\text{FO}$?
Database-theoretic Dependencies

Dependence Logic $FO(\equiv(\cdot, \cdot))$: $M \models_X \equiv(\vec{y}, \vec{z})$ iff for all $s, s' \in X$, $s(\vec{y}) = s'(\vec{y}) \Rightarrow s(\vec{z}) = s'(\vec{z})$;

Exclusion Logic $FO(\lvert)$: $M \models_X \vec{y} \lvert \vec{z}$ iff $X(\vec{y}) \cap X(\vec{z}) = \emptyset$;

Inclusion Logic $FO(\subseteq)$: $M \models_X \vec{y} \subseteq \vec{z}$ iff $X(\vec{y}) \subseteq X(\vec{z})$;

Independence Logic $FO(\bot)$: $M \models_X \vec{y} \bot \vec{z}$ iff $X(\vec{y}\vec{z}) = X(\vec{y}) \times X(\vec{z})$.

First Order?

These dependencies are first-order (as properties of relations), but the resulting logics are not: for example, Independence Logic = $\Sigma_1^1$ and Inclusion Logic = $GFP^+$.

The Question

Under which conditions is $FO(\mathcal{D})$ equivalent to $FO$?
Let’s be more precise...

**Generalized Dependencies (Kuusisto 2013)**

A *dependency* $D$ of arity $k$ is a class, closed under isomorphisms, of models $(\text{Dom}(M), R)$ for $R$ $k$-ary.

$$M \models_X D \vec{y} \iff (\text{Dom}(M), X(\vec{y})) \in D.$$  

**First-Order Dependencies**

$D$ is *first order* if there is a F.O. sentence $\psi^*(R)$ s.t.

$$(\text{Dom}(M), R) \in D \iff (\text{Dom}(M), R) \models \psi^*(R).$$
A Classification of Dependencies

Three Properties of Dependencies

Empty Team: $M \models_{\emptyset} D\vec{y}$ for all $M$ and all $\vec{y}$;

Downwards Closure: If $M \models_X D\vec{y}$ and $Y \subseteq X$ then $M \models_Y D\vec{y}$;

Union Closure: If $M \models_{X_i} D\vec{y}$ for all $i \in I$ then $M \models_{\bigcup_i X_i} D\vec{y}$.

Observation

All union closed dependencies and all nontrivial downwards closed dependencies satisfy the Empty Team Property.

Another observation

These properties are preserved by Team Semantics: for example, if $D$ has downwards closure and $\phi \in FO(D)$ then $\phi$ has downwards closure, that is,

$$M \models_X \phi, Y \subseteq X \Rightarrow M \models_Y \phi.$$
<table>
<thead>
<tr>
<th>Dependency</th>
<th>Empty Team?</th>
<th>Downwards Cl.?</th>
<th>Union Cl.?</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(y, z)$</td>
<td>YES</td>
<td>YES</td>
<td>NO</td>
</tr>
<tr>
<td>$y \mid z$</td>
<td>YES</td>
<td>YES</td>
<td>NO</td>
</tr>
<tr>
<td>$y \subseteq z$</td>
<td>YES</td>
<td>NO</td>
<td>YES</td>
</tr>
<tr>
<td>$y \perp z$</td>
<td>YES</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>NE</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
</tr>
</tbody>
</table>
Some Known Results

Three families of dependencies

1. $\mathcal{D}^{ET} = \{ D : D \text{ F.O. definable, empty team prop.} \}$;
2. $\mathcal{D}^\downarrow = \{ D : D \text{ F.O. def, downwards closed, empty team pr.} \}$;
3. $\mathcal{D}^\cup = \{ D : D \text{ F.O. def., union closed } \}$.

Some Results (wrt formulas)

Kontinen, Väänänen 2009: Dependence Logic = $FO(\mathcal{D}^\downarrow)$;
Galliani 2012: Independence Logic = $FO(\mathcal{D}^{ET})$;
Galliani, Hella 2013: Inclusion Logic = $FO(\mathcal{D}^\cup)$.
Some Known Results

Three families of dependencies

1. $\mathcal{D}^{ET} = \{ D : D \text{ F.O. definable, empty team prop.} \}$;
2. $\mathcal{D}^{\downarrow} = \{ D : D \text{ F.O. def, downwards closed, empty team pr.} \}$;
3. $\mathcal{D}^{U} = \{ D : D \text{ F.O. def., union closed} \}$.

Some More Results (wrt sentences)

Väänänen 2007: Dependence Logic = $\Sigma^1_1$;
Grädel, Väänänen 2012: Independence Logic = $\Sigma^1_1$;
Galliani, Hella 2013: Inclusion Logic = $GFP^+$. 
Relations between Dependence Families (formulas)

\[
\begin{align*}
fo(D^E \cup) & \subset fo(D^T) \\
fo(D^E \cup) & \supset fo(D^T) \\
fo(D^T) & \subset fo(D^E) \\
fo(D^T) & \supset fo(D^E)
\end{align*}
\]
Relations between Dependence Families (Sentences)

\[
FO(\mathcal{D}^E), \quad FO(\mathcal{D}^F) \quad \equiv \Sigma^1
\]

\[
\bigcup
\]

\[
FO(\mathcal{D}^U) \quad \equiv \text{GFP}^+
\]

\[
\bigcup
\]

\[
FO \quad \equiv \text{FO}
\]
1. Team Semantics and Dependencies
2. Upwards Closed Dependencies
3. Boundedness
So far, we identified families of dependencies $\mathcal{D}$ such that $\text{FO}(\mathcal{D})$ is stronger than First Order Logic.

But can we find (non-trivial) families $\mathcal{D}$ such that $\text{FO}(\mathcal{D}) = \text{FO}$?

How much can we get away with adding to Team Semantics before “jumping” beyond First Order Logic?

**Strongly First Order Dependencies**

Let $\mathcal{D} = \{D_1, D_2, \ldots \}$ be a family of first order definable dependencies. We say that $\mathcal{D}$ is strongly first order if every sentence of $\text{FO}(\mathcal{D})$ is equivalent to some first order sentence.

**Open Problem**

Find necessary and sufficient conditions for $\mathcal{D}$ to be strongly first order.
So far, we identified families of dependencies $\mathcal{D}$ such that $FO(\mathcal{D})$ is stronger than First Order Logic.

But can we find (non-trivial) families $\mathcal{D}$ such that $FO(\mathcal{D}) = FO$?

How much can we get away with adding to Team Semantics before “jumping” beyond First Order Logic?

**Strongly First Order Dependencies**

Let $\mathcal{D} = \{D_1, D_2, \ldots \}$ be a family of first order definable dependencies. We say that $\mathcal{D}$ is *strongly first order* if every sentence of $FO(\mathcal{D})$ is equivalent to some first order sentence.

**Open Problem**

Find necessary and sufficient conditions for $\mathcal{D}$ to be strongly first order.
**Constancy Logic**

*Constancy Logic FO(=(*)) is the fragment of Dependence Logic containing only constancy atoms =((∅, ⃗z)) (also written just =(⃗z)).*

**Galliani 2012**

Let $C$ be the set of all constancy dependencies. Then $C$ is strongly first order.

Can we find something more general?
Constancy Logic

Constancy Logic FO(\(\equiv(\cdot)\)) is the fragment of Dependence Logic containing only constancy atoms \(\equiv(\emptyset, \vec{z})\) (also written just \(\equiv(\vec{z})\)).

Galliani 2012

Let \(\mathcal{C}\) be the set of all constancy dependencies. Then \(\mathcal{C}\) is strongly first order.

Can we find something more general?
Upwards Closed Dependencies

We say that a dependency $D$ is *upwards closed* if

$$(\text{Dom}(M), R) \in D, R \subseteq S \Rightarrow (\text{Dom}(M), S) \in D$$

or, equivalently, if $M \models_X D\vec{y}, X \subseteq Y \Rightarrow M \models_Y Dy$.

**Examples**

**Non-Emptiness:** $M \models_X \text{NE}$ iff $X \neq \emptyset$;

**$n$-bigness:** $M \models_X |\vec{y}| \geq n$ iff $|X(\vec{y})| \geq n$;

**Totality:** $M \models_X \text{All}(\vec{y})$ iff $X(\vec{y}) = \text{Dom}(M)|\vec{y}|$;

**Intersection:** $M \models_X \vec{y} \not\in \vec{z}$ iff $X(\vec{y}) \cap X(\vec{z}) \neq \emptyset$;

**$\kappa$-bigness:** For $\kappa$ infinite cardinal, $M \models_X |\vec{y}| \geq \kappa$ iff $|X(\vec{y})| \geq \kappa$. 

---

Pietro Galliani

Strongly First-Order Dependencies in Team Semantics
On Upwards Closed Dependencies

**Upwards Closure is Not Preserved**

If $\mathcal{D}$ is a family of upwards closed dependencies then in general $\text{FO}(\mathcal{D})$ is not upwards closed: $M \models_X \phi$, $X \subseteq Y \not\Rightarrow M \models_Y \phi$.

**Theorem (Galliani 2013)**

If $\mathcal{D}$ is a family of first order upwards closed dependencies and constancy atoms then $\mathcal{D}$ is strongly first order: $\text{FO}(\mathcal{D}) = \text{FO}$ over sentences.

**Corollary**

The set of the contradictory negations of inclusion, exclusion, dependence and independence atoms is strongly first order.
I will sketch the proof of the case *without* constancy atoms. Extending the proof to constancy atoms is easy.

### An Observation

**Why might we go beyond first order logic?**

**Disjunction:** \[ M \models_X \phi \lor \psi \iff \exists Y, Z \text{ s.t. } X = Y \cup Z, \; M \models_Y \phi, \; M \models_Z \psi; \]

**Existential:** \[ M \models_X \exists v \phi \iff \exists H \text{ s.t. } M \models_X[H/v] \phi. \]

### A proof plan

Suppose that all dependencies are upwards closed. Find a way to evaluate disjunction and existential quantification *without* quantification over teams or functions, then the result follows.
Flattening

Let $\phi \in FO(D)$. Then $\phi^f$ is the first order formula obtained by replacing any dependence $D\vec{y}$ with $\top$.

Note

If $M \models_X \phi$ then $M \models_X \phi^f$ and, for all $s \in X$, $M \models_s \phi^f$. 
The Flattening Lemma

If all $D$ in $\phi$ are upwards closed, $M \models_X \phi$, $X \subseteq Y$ and $M \models_s \phi^f$ for all $s \in Y$ then $M \models_Y \phi$. 

\[ M \models X \phi \]
The Flattening Lemma

Flattening Lemma

If all $D$ in $\phi$ are upwards closed, $M \models_X \phi$, $X \subseteq Y$ and $M \models_s \phi^f$ for all $s \in Y$ then $M \models_Y \phi$. 

\[
\forall s \in Y, M \models_s \phi^f
\]
The Flattening Lemma

If all $D$ in $\phi$ are upwards closed, $M \models_X \phi$, $X \subseteq Y$ and $M \models_s \phi^f$ for all $s \in Y$ then $M \models_Y \phi$. 

\[ \forall s \in Y, M \models_s \phi^f \]
Evaluating Disjunctions Greedily

\[ M \models_X \phi \vee \psi \text{ iff } \exists Y, Z \text{ s.t. } X = Y \cup Z, M \models_Y \phi, M \models_Z \psi \]

Let \( X_1 = \{ s \in X, M \models_s \phi^f \}, X_2 = \{ s \in X, M \models_s \psi^f \}. \) Then

\[ M \models_X \phi \vee \psi \text{ iff } X = X_1 \cup X_2, M \models_{X_1} \phi \text{ and } M \models_{X_2} \psi. \]
$M \models X \phi \lor \psi$ iff $\exists Y, Z$ s.t. $X = Y \cup Z$, $M \models Y \phi$, $M \models Z \psi$

Let $X_1 = \{ s \in X, M \models s \phi \}$, $X_2 = \{ s \in X, M \models s \psi \}$. Then

$M \models X \phi \lor \psi$ iff $X = X_1 \cup X_2$, $M \models X_1 \phi$ and $M \models X_2 \psi$. 
Evaluating Disjunctions Greedily

\[ M \models X \phi \lor \psi \iff \exists Y, Z \text{ s.t. } X = Y \cup Z, M \models Y \phi, M \models Z \psi \]

Let \( X_1 = \{ s \in X, M \models s \phi^f \} \), \( X_2 = \{ s \in X, M \models s \psi^f \} \). Then

\[ M \models X \phi \lor \psi \iff X = X_1 \cup X_2, M \models X_1 \phi \text{ and } M \models X_2 \psi. \]
Evaluating Disjunctions Greedily

\[ M \models_X \phi \lor \psi \text{ iff } \exists Y, Z \text{ s.t. } X = Y \cup Z, \ M \models_Y \phi, \ M \models_Z \psi \]

Let \( X_1 = \{ s \in X, M \models_s \phi^f \} \), \( X_2 = \{ s \in X, M \models_s \psi^f \} \). Then

\[ M \models_X \phi \lor \psi \text{ iff } X = X_1 \cup X_2, \ M \models_{X_1} \phi \text{ and } M \models_{X_2} \psi. \]
$M \models_X \phi \lor \psi$ iff $\exists Y, Z$ s.t. $X = Y \cup Z$, $M \models_Y \phi$, $M \models_Z \psi$

Let $X_1 = \{ s \in X, M \models_s \phi^f \}$, $X_2 = \{ s \in X, M \models_s \psi^f \}$. Then

$M \models_X \phi \lor \psi$ iff $X = X_1 \cup X_2$, $M \models_{X_1} \phi$ and $M \models_{X_2} \psi$. 

Evaluating Disjunctions Greedily
Evaluating Disjunctions Greedily

\[ M \models_X \phi \lor \psi \iff \exists Y, Z \text{ s.t. } X = Y \cup Z, \ M \models_Y \phi, \ M \models_Z \psi \]

Let \( X_1 = \{ s \in X, M \models_s \phi^f \} \), \( X_2 = \{ s \in X, M \models_s \psi^f \} \). Then

\[ M \models_X \phi \lor \psi \iff X = X_1 \cup X_2, \ M \models_{X_1} \phi \text{ and } M \models_{X_2} \psi. \]
Evaluating Disjunctions Greedily

\[ M \models_X \phi \lor \psi \iff \exists Y, Z \text{ s.t. } X = Y \cup Z, M \models_Y \phi, M \models_Z \psi \]

Let \( X_1 = \{ s \in X, M \models_s \phi^f \} \), \( X_2 = \{ s \in X, M \models_s \psi^f \} \). Then

\[ M \models_X \phi \lor \psi \iff X = X_1 \cup X_2, M \models_{X_1} \phi \text{ and } M \models_{X_2} \psi. \]
$M \models_X \phi \lor \psi$ iff $\exists Y, Z \text{ s.t. } X = Y \cup Z, M \models_Y \phi, M \models_Z \psi$

Let $X_1 = \{ s \in X, M \models_s \phi^f \}$, $X_2 = \{ s \in X, M \models_s \psi^f \}$. Then

$M \models_X \phi \lor \psi$ iff $X = X_1 \cup X_2, M \models_{X_1} \phi$ and $M \models_{X_2} \psi$.
$M \models_{X} \exists v \phi \iff \exists H \text{ s.t. } M \models_{X[H/v]} \phi$

For all $s \in X$, let $K(s) = \{m \in \text{Dom}(M) : M \models_{s[m/v]} \phi^f\}$. Then $M \models_{X} \exists v \phi \iff$

- For all $s \in X$, $K(s) \neq \emptyset$;
- For $Y = X[K/v] = \{s[m/v] : s \in X, m \in K(s)\}$, $M \models_{Y} \psi$. 

Pietro Galliani

Strongly First-Order Dependencies in Team Semantics
1. Team Semantics and Dependencies
2. Upwards Closed Dependencies
3. Boundedness
A dependency $D$ is $k$-bounded (in $M$) if whenever $M \models_X D \vec{v}$, there exists a $Y \subseteq X$ with $|Y| \leq k$ such that $M \models_Y D \vec{v}$.
A dependency $D$ is $k$-bounded (in $M$) if whenever $M \models_X D\vec{v}$, there exists a $Y \subseteq X$ with $|Y| \leq k$ such that $M \models_Y D\vec{v}$. 
A dependency $D$ is $k$-bounded (in $M$) if whenever $M \models_X D \vec{v}$, there exists a $Y \subseteq X$ with $|Y| \leq k$ such that $M \models_Y D \vec{v}$. 
Boundedness

A dependency \( D \) is \( k \)-bounded (in \( M \)) if whenever \( M \models_X D\vec{v} \), there exists a \( Y \subseteq X \) with \( |Y| \leq k \) such that \( M \models_Y D\vec{v} \).

Examples

Non-Emptiness: \( M \models_X NE \) iff \( X \neq \emptyset \):
A dependency $D$ is $k$-bounded (in $M$) if whenever $M \models_X D\vec{v}$, there exists a $Y \subseteq X$ with $|Y| \leq k$ such that $M \models_Y D\vec{v}$.

**Examples**

**Non-Emptiness:** $M \models_X \text{NE}$ iff $X \neq \emptyset$: 1-bounded;
Boundedness

A dependency $D$ is $k$-bounded (in $M$) if whenever $M \models_X D \vec{v}$, there exists a $Y \subseteq X$ with $|Y| \leq k$ such that $M \models_Y D \vec{v}$.

Examples

Non-Emptiness: $M \models_X \text{NE}$ iff $X \neq \emptyset$: 1-bounded;

$n$-bigness: $M \models_X |\vec{y}| \geq n$ iff $|X(\vec{y})| \geq n$: 
Boundedness

A dependency $D$ is $k$-bounded (in $M$) if whenever $M \models_X D \vec{v}$, there exists a $Y \subseteq X$ with $|Y| \leq k$ such that $M \models_Y D \vec{v}$.

Examples

Non-Emptiness: $M \models_X \text{NE}$ iff $X \neq \emptyset$: 1-bounded;

$n$-bigness: $M \models_X |\vec{y}| \geq n$ iff $|X(\vec{y})| \geq n$: $n$-bounded;
Boundedness

A dependency $D$ is $k$-bounded (in $M$) if whenever $M \models_X D \vec{v}$, there exists a $Y \subseteq X$ with $|Y| \leq k$ such that $M \models_Y D \vec{v}$.

Examples

Non-Emptiness: $M \models_X \text{NE}$ iff $X \neq \emptyset$: 1-bounded;

$n$-bigness: $M \models_X |\vec{y}| \geq n$ iff $|X(\vec{y})| \geq n$: $n$-bounded;

Totality: $M \models_X \text{All}(\vec{y})$ iff $X(\vec{y}) = \text{Dom}(M)|\vec{y}|$:
Boundedness

A dependency $D$ is $k$-bounded (in $M$) if whenever $M \models_X D\vec{v}$, there exists a $Y \subseteq X$ with $|Y| \leq k$ such that $M \models_Y D\vec{v}$.

Examples

Non-Emptiness: $M \models_X \text{NE}$ iff $X \neq \emptyset$: 1-bounded;

$n$-bigness: $M \models_X |\vec{y}| \geq n$ iff $|X(\vec{y})| \geq n$: $n$-bounded;

Totality: $M \models_X \text{All}(\vec{y})$ iff $X(\vec{y}) = \text{Dom}(M|\vec{y}|)$: $|M|\vec{y}|$-bounded;
Boundedness

A dependency $\mathbf{D}$ is $k$-bounded (in $M$) if whenever $M \models_\mathbf{X} \mathbf{D} \vec{v}$, there exists a $Y \subseteq X$ with $|Y| \leq k$ such that $M \models_Y \mathbf{D} \vec{v}$.

Examples

Non-Emptiness: $M \models_\mathbf{X} \text{NE}$ iff $X \neq \emptyset$: 1-bounded;

$n$-bigness: $M \models_\mathbf{X} |\vec{y}| \geq n$ iff $|X(\vec{y})| \geq n$: $n$-bounded;

Totality: $M \models_\mathbf{X} \text{All}(\vec{y})$ iff $X(\vec{y}) = \text{Dom}(M)|\vec{y}|$: $|M||\vec{y}|$-bounded;

Intersection: $M \models_\mathbf{X} \vec{y} \not\parallel \vec{z}$ iff $X(\vec{y}) \cap X(\vec{z}) \neq \emptyset$: 
A dependency $D$ is $k$-bounded (in $M$) if whenever $M \models_X D \vec{v}$, there exists a $Y \subseteq X$ with $|Y| \leq k$ such that $M \models_Y D \vec{v}$.

**Examples**

- **Non-Emptyness**: $M \models_X \text{NE}$ iff $X \neq \emptyset$: 1-bounded;
- **$n$-bigness**: $M \models_X |\vec{y}| \geq n$ iff $|X(\vec{y})| \geq n$: $n$-bounded;
- **Totality**: $M \models_X \text{All}(\vec{y})$ iff $X(\vec{y}) = \text{Dom}(M)|\vec{y}|$: $|M||\vec{y}|$-bounded;
- **Intersection**: $M \models_X \vec{y} \not\subseteq \vec{z}$ iff $X(\vec{y}) \cap X(\vec{z}) \neq \emptyset$: 2-bounded.
Boundedness Theorem

Let \( \mathcal{D} \) be a family of upwards closed dependencies, let \( \phi \in FO(\mathcal{D}) \) contain upwards closed dependency instances \( D_1 \vec{y}_1 \ldots D_k \vec{y}_k \) which are \( n_1-\), \ldots, \( n_k- \) bounded. Then \( \phi \) is \( n = n_1 + \ldots + n_k \)-bounded:

\[
M \models_X \phi \Rightarrow \exists Y \subseteq X, |Y| \leq n, \text{ s.t. } M \models_Y \phi.
\]

Proof.

Proof (sketch) First show that \( \phi \) can be put in the form

\[
\exists z_1 \ldots z_t (\mathcal{=} (z_1) \land \ldots \land \mathcal{=} (z_t) \land \psi),
\]

where \( \psi \) contains no constancy atoms. Then proceed by induction on \( \psi \).
Corollary

If $\mathcal{D}$ is a family of \textit{constantly-bounded} upwards closed dependencies. Then $\mathsf{All}(y)$ is not definable in $\mathsf{FO}(\mathcal{D})$: there is no formula $\phi(y)$ such that $M \models_x \phi(y)$ iff $M \models_x \mathsf{All}(y)$.

Corollary

$\mathsf{All}(y_1 \ldots y_n)$ is \textit{not} definable in terms of $\mathsf{All}(y_1 \ldots y_{n-1})$.

Corollary

If $\mathcal{D}$ is $k$-bounded and upwards closed and $\phi(y) \in \mathsf{FO}(\mathcal{D})$ characterizes $n$-bigness then $\phi$ contains at least $\lceil \frac{n}{k} \rceil$ instances of $\mathcal{D}$. 
Corollary

If $\mathcal{D}$ is a family of \textit{constantly-bounded} upwards closed dependencies. Then $\text{All}(y)$ is not definable in $FO(= (\cdot), \mathcal{D})$: there is no formula $\phi(y)$ such that $M \models x \phi(y)$ iff $M \models x \text{All}(y)$.

Corollary

$\text{All}(y_1 \ldots y_n)$ is \textit{not} definable in terms of $\text{All}(y_1 \ldots y_{n-1})$.

Corollary

If $\mathcal{D}$ is $k$-bounded and upwards closed and $\phi(y) \in FO(= (\cdot), \mathcal{D})$ characterizes $n$-bigness then $\phi$ contains at least $\lceil \frac{n}{k} \rceil$ instances of $\mathcal{D}$. 

Pietro Galliani

Strongly First-Order Dependencies in Team Semantics
Corollary
If $\mathcal{D}$ is a family of constantly-bounded upwards closed dependencies. Then $\forall y$ is not definable in $FO(\equiv(\cdot), \mathcal{D})$: there is no formula $\phi(y)$ such that $M \models x \phi(y)$ iff $M \models x \forall y$.

Corollary
$\forall y_1 \ldots y_n$ is not definable in terms of $\forall y_1 \ldots y_{n-1}$.

Corollary
If $\mathcal{D}$ is $k$-bounded and upwards closed and $\phi(y) \in FO(\equiv(\cdot), \mathcal{D})$ characterizes $n$-bigness then $\phi$ contains at least $\lceil \frac{n}{k} \rceil$ instances of $\mathcal{D}$. 
Conclusions

What we have

- Team Semantics is a rich semantical framework not only for higher-order logics, but also for seeing first-order logic from a different perspective!
- We characterized a class of dependencies $\mathcal{D}$ such that $\text{FO}(\mathcal{D}) = \text{FO}$ wrt sentences;
- Non-definability results through boundedness.

What we still need

- Necessary and sufficient conditions for $\text{FO}(\mathcal{D}) = \text{FO}$?
- Is there a single dependency $\mathcal{D}$ such that $\text{FO}(\mathcal{D}) = \text{FO}(\mathcal{D}^\uparrow)$?
- Which properties of teams are definable in $\text{FO(A11)}$, $\text{FO(NE)}$, and other such logics?
Conclusions

What we have

- Team Semantics is a rich semantical framework not only for higher-order logics, but also for seeing first-order logic from a different perspective!
- We characterized a class of dependencies $\mathcal{D}$ such that $FO(\mathcal{D}) = FO$ wrt sentences;
- Non-definability results through boundedness.

What we still need

- Necessary and sufficient conditions for $FO(\mathcal{D}) = FO$?
- Is there a single dependency $\mathcal{D}$ such that $FO(\mathcal{D}) = FO(\mathcal{D}^\uparrow)$?
- Which properties of teams are definable in $FO(\text{All})$, $FO(\text{NE})$, and other such logics?