Dependence and Independence in Logic

Erich Grädel

Tutorial at Logic Colloquium 2015, Helsinki, August 2015

Part I: Dependence. Independence, Team Semantics

- Motivation and historical remarks
- Dependence and independence as atomic properties
- Logics of dependence and independence. Team semantics
- Expressive power
- Negation

Dependence and independence

What does it mean that "x depends on y" or that "x and y are independent" ?

Dependence and independence

What does it mean that "x depends on y" or that "x and y are independent" ?

Compare this to statements of a more common form such as "x divides y".

Dependence and independence

What does it mean that "x depends on y" or that "x and y are independent" ?

Compare this to statements of a more common form such as "x divides y".

To make sense of this we fix a structure $\mathfrak{A}$ in which the notion of divisibility is well-defined and an assignment $s : \{x, y\} \to A$ and use Tarski-style semantics to determine whether $\mathfrak{A} \models_s$ "x divides y".

Dependence and independence

What does it mean that "x depends on y" or that "x and y are independent" ?

Compare this to statements of a more common form such as "x divides y".

To make sense of this we fix a structure $\mathfrak{A}$ in which the notion of divisibility is well-defined and an assignment $s : \{x, y\} \to A$ and use Tarski-style semantics to determine whether $\mathfrak{A} \models_s$ "x divides y".

Dependence and independence are notions of a different kind! They manifest themselves not in single assignments, but only in larger amounts of data:

- tables or relations
- sets of plays in a game (e.g. strategies)
- sets of assignments.

Dependence and independence

What does it mean that "x depends on y" or that "x and y are independent" ?

Compare this to statements of a more common form such as "x divides y".

To make sense of this we fix a structure $\mathfrak{A}$ in which the notion of divisibility is well-defined and an assignment $s : \{x, y\} \to A$ and use Tarski-style semantics to determine whether $\mathfrak{A} \models_s$ "x divides y".

Dependence and independence are notions of a different kind! They manifest themselves not in single assignments, but only in larger amounts of data:

- tables or relations
- sets of plays in a game (e.g. strategies)
- sets of assignments.

A set of assigments (all with the same domain of variables) is called a team.

Logics of dependence and independence

Henkin, Enderton, Walkoe, . . . : partially ordered (or Henkin-) quantifiers

Blass and Gurevich: correspondence to Σ^1_1 (and thus NP)

Logics of dependence and independence

Henkin, Enderton, Walkoe, . . . : partially ordered (or Henkin-) quantifiers
Blass and Gurevich: correspondence to Σ^1_1 (and thus NP)

Hintikka and Sandu: Independence-friendly (IF) logic with explicit dependencies of quantifiers on each other

Semantics in terms of games with imperfect information

Logics of dependence and independence

Henkin, Enderton, Walkoe, . . . : partially ordered (or Henkin-) quantifiers
Blass and Gurevich: correspondence to Σ^1_1 (and thus NP)

Hintikka and Sandu: Independence-friendly (IF) logic with explicit dependencies of quantifiers on each other

Semantics in terms of games with imperfect information

Claim: Impossibility of model-theoretic (compositional) semantics
never made precise, never proved

Logics of dependence and independence

Henkin, Enderton, Walkoe, . . . : partially ordered (or Henkin-) quantifiers
Blass and Gurevich: correspondence to Σ^1_1 (and thus NP)

Hintikka and Sandu: Independence-friendly (IF) logic with explicit dependencies of quantifiers on each other

Semantics in terms of games with imperfect information

Claim: Impossibility of model-theoretic (compositional) semantics
never made precise, never proved

Hodges: model-theoretic semantics for IF-logic

Difference to Tarski semantics: a formula is not evaluated against a single assignment but against a set of assignments

This kind of semantics is an important achievement of independent interest. The full potential of this innovation has still not been fully appreciated.

Henkin quantifiers

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

Intuitively, this says that for all x, u there exist y, v such that $Pxyuv$ holds, and moreover, the choice of y only depends on the value of x and the choice of v only depends on the value of u.

Henkin quantifiers

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

Intuitively, this says that for all x, u there exist y, v such that $Pxyuv$ holds, and moreover, the choice of y only depends on the value of x and the choice of v only depends on the value of u.

Precise semantics can be given either in terms of Skolem functions, or in terms of games of imperfect information.

Henkin quantifiers

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

Intuitively, this says that for all x, u there exist y, v such that $Pxyuv$ holds, and moreover, the choice of y only depends on the value of x and the choice of v only depends on the value of u.

Precise semantics can be given either in terms of Skolem functions, or in terms of games of imperfect information.

$(A, P) \models \varphi$ if there exist functions $f, g : A \to A$ such that $P(a, fa, c, gc)$ holds for all $a, c \in A$.

Henkin quantifiers and NP

Simple formulae with Henkin quantifiers express NP-complete problems

A graph $G = (V, E)$ is 3-colourable if, and only if,

$$G \models \begin{pmatrix} \forall x & \exists y \\ \forall u & \exists v \end{pmatrix} \Big(y, v \in \{0,1,2\} \wedge (x = u \to y = v) \wedge (Exu \to y \neq v) \Big)$$

Theorem (Blass, Gurevich)

Henkin quantfiers over first-order formulae capture NP

Independence-friendly logic

IF-logic is first-order logic where quantifiers are annotated by independencies

Quantification rule: If φ is a formula, x is a variable, and W is a finite set of variables, then the expressions $(\exists x/W)\varphi$ and $(\forall x/W)\varphi$ are also formulae.

Game-theoretic semantics: In the evaluation game for $(Qx/W)\varphi$, the value for x must be chosen independently from the the values of the variables in W.

At two positions $((Qx/W)\varphi, s)$ and $((Qx/W)\varphi, s')$ such that $s(y) \neq s'(y)$ only for variables in W, the same value for x must be chosen.

IF-logic and existential second-order logic

A graph $G = (V, E)$ admits a perfect matching if, and only if,

$$G \models \forall x \forall y (\exists u / \{y\})(\exists v / \{x, u\}) \Big((x = y \to u = v) \wedge (u = y \to v = x) \wedge Exu \Big)$$

IF-logic and existential second-order logic

A graph $G = (V, E)$ admits a perfect matching if, and only if,

$$G \models \forall x \forall y (\exists u/\{y\})(\exists v/\{x,u\})\Big((x = y \to u = v) \wedge (u = y \to v = x) \wedge Exu\Big)$$

Skolem semantics. Replace existential quantifiers $(\exists x/W)$ by Skolem functions whose arguments are the variables outside W.

IF-logic and existential second-order logic

A graph $G = (V, E)$ admits a perfect matching if, and only if,

$$G \models \forall x \forall y (\exists u / \{y\})(\exists v / \{x, u\}) \Big((x = y \to u = v) \wedge (u = y \to v = x) \wedge Exu \Big)$$

Skolem semantics. Replace existential quantifiers $(\exists x / W)$ by Skolem functions whose arguments are the variables outside W.

The formula for the perfect matching becomes

$$(\exists f)(\exists g) \forall x \forall y \Big((x = y \to fx = gy) \wedge (fx = y \to gy = x) \wedge Exfx \Big)$$

which is equivalent to $(\exists f) \forall x (ffx = x \wedge Exfx)$

IF-logic and existential second-order logic

A graph $G = (V, E)$ admits a perfect matching if, and only if,

$$G \models \forall x \forall y (\exists u / \{y\})(\exists v / \{x, u\}) \Big((x = y \to u = v) \wedge (u = y \to v = x) \wedge Exu \Big)$$

Skolem semantics. Replace existential quantifiers $(\exists x / W)$ by Skolem functions whose arguments are the variables outside W.

The formula for the perfect matching becomes

$$(\exists f)(\exists g) \forall x \forall y \Big((x = y \to fx = gy) \wedge (fx = y \to gy = x) \wedge Exfx \Big)$$

which is equivalent to $(\exists f) \forall x (ffx = x \wedge Exfx)$

Theorem. In expressive power, IF-logic is equivalent to existential second-order logic and thus captures NP on finite structures.

Logics of dependence and independence

Väänänen: Dependence logic

rather than stating dependencies or independencies as annotations of quantifiers, treat dependence as an atomic statement

model-theoretic semantics in terms of sets of assignments (team semantics)

Logics of dependence and independence

Väänänen: Dependence logic

rather than stating dependencies or independencies as annotations of quantifiers, treat dependence as an atomic statement

model-theoretic semantics in terms of sets of assignments (team semantics)

Grädel, Väänänen: Independence logic

based on independence rather than dependence
dependence as a special case of a general form of independence

Logics of dependence and independence

Väänänen: Dependence logic

rather than stating dependencies or independencies as annotations of quantifiers, treat dependence as an atomic statement

model-theoretic semantics in terms of sets of assignments (team semantics)

Grädel, Väänänen: Independence logic

based on independence rather than dependence
dependence as a special case of a general form of independence

Galliani: Logics based on other dependency properties

Dependence atoms

Dependence atoms are expressions $=(\overline{x}, y)$.

Semantics: Let $\mathfrak{A}$ be a structure and X a team of assignments $s : V \to A$.

$\mathfrak{A} \models_X =(\overline{x}, y)$ if y depends on $\overline{x}$ in $\mathfrak{A}$ and X.

This means that for all $s, s' \in X$,

$$\bigwedge_{i=1}^{n} s(x_i) = s'(x_i) \implies s(y) = s'(y)$$

Dependence atoms

Dependence atoms are expressions $=(\overline{x}, y)$.

Semantics: Let $\mathfrak{A}$ be a structure and X a team of assignments $s : V \to A$.

$\mathfrak{A} \models_X =(\overline{x}, y)$ if y depends on $\overline{x}$ in $\mathfrak{A}$ and X.

This means that for all $s, s' \in X$,

$$\bigwedge_{i=1}^{n} s(x_i) = s'(x_i) \implies s(y) = s'(y)$$

Expanded form of dependence atoms $=(\overline{x}, \overline{y})$ saying that all variables of $\overline{y}$ depend on $\overline{x}$.

Thus dependence is understood as functional dependence, which is the strongest form of dependence. The values of $x_1, \ldots x_n$ in X determine the value of y as surely as x and y determine $x + y$ and $x \cdot y$ in elementary arithmetic. Weaker forms of dependence are definable from that.

Armstrong's axioms for dependence

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

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

(3) If $=(\overline{x}, \overline{y})$ and $\overline{x}'$ and $\overline{y}'$ are permutations of $\overline{x}$ and $\overline{y}$, then $=(\overline{x}', \overline{y}')$.
Functional dependence does not look at the order of the variables.

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

Completeness: If T is a finite set of dependence atoms of the form $=(\overline{x}, \overline{y})$ for various $\overline{x}$ and $\overline{y}$, then $=(\overline{x}, \overline{y})$ follows from T according to Armstrong's rules if, and only if, every team that satisfies T also satisfies $=(\overline{x}, \overline{y})$.

Independence: informal discussion

Independence is a more complicated notion than dependence.

Strongest form of logical independence of x and y:

- Every conceivable pattern of values for (x, y) occurs.
- Knowing one of x and y gives no information about the other.

Caution. Probability theory has its own concept of independence: two random variables are independent if observing one does not affect the probabilities of the other. Logical independence is in harmony with probabilistic independence, but does not pay attention to how often a pattern occurs.

Example for independence

Suppose you want gather experimental data by throwing balls of various size and masses from the Leaning Tower of Pisa to observe how size and mass influence the time of descent. Setting up the experiment you may want to make sure that

The size of the ball is independent from its mass

Example for independence

Suppose you want gather experimental data by throwing balls of various size and masses from the Leaning Tower of Pisa to observe how size and mass influence the time of descent. Setting up the experiment you may want to make sure that

The size of the ball is independent from its mass

How to make sure of this? Vary sizes and masses so that if one size is chosen for one mass it is also chosen for all other masses (and vice versa). This would eliminate any dependence between size and mass and the experiment would genuinely tell us something relevant about the time of descent.

Independence atoms: simple case

Definition. A team X satisfies the atom $x \perp y$ if

$$(\forall s, s' \in X)(\exists s'' \in X)(s''(x) = s(x) \wedge s''(y) = s'(y))$$

Independence atoms: simple case

Definition. A team X satisfies the atom $x \perp y$ if

$$(\forall s, s' \in X)(\exists s'' \in X)(s''(x) = s(x) \wedge s''(y) = s'(y))$$

Suppose you know X and you know that s is some assignment in X. You want to gather information about $s(y)$. What you know is that $s(y) \in \{a : a = s'(y) \text{ for some } s' \in X\}$.

Independence atoms: simple case

Definition. A team X satisfies the atom $x \perp y$ if

$$(\forall s, s' \in X)(\exists s'' \in X)(s''(x) = s(x) \wedge s''(y) = s'(y))$$

Suppose you know X and you know that s is some assignment in X.
You want to gather information about $s(y)$. What you know is that
$s(y) \in \{a : a = s'(y) \text{ for some } s' \in X\}$.

Suppose that I tell you $s(x)$ where $x \perp y$.
You cannot infer anything new about $s(y)$. Indeed, for all potential values a for $s(y)$ there is an assignment $s'' \in X$ with $s''(x) = s(x)$ and $s''(y) = a$.

Constants

Consider a team of data obtained by throwing objects of the same size but different masses from the leaning tower of Pisa.

Galileo: The time of descent is independent of the mass.

Constants

Consider a team of data obtained by throwing objects of the same size but different masses from the leaning tower of Pisa.

Galileo: The time of descent is independent of the mass.

Special theory of relativity:

Einstein: The speed of light is independent of the observer's state of motion.

Constants

Consider a team of data obtained by throwing objects of the same size but different masses from the leaning tower of Pisa.

Galileo: The time of descent is independent of the mass.

Special theory of relativity:

Einstein: The speed of light is independent of the observer's state of motion.

A special case of total independence is when one of the variables is a constant !

Constants

Consider a team of data obtained by throwing objects of the same size but different masses from the leaning tower of Pisa.

Galileo: The time of descent is independent of the mass.

Special theory of relativity:

Einstein: The speed of light is independent of the observer's state of motion.

A special case of total independence is when one of the variables is a constant !

Proposition. A constant variable is independent from every other variable including itself: $=(x) \iff x \perp x$

Independence atoms: general case

The independence atom $x \perp y$, and also its extension to independency atoms $\overline{x} \perp \overline{y}$ on tuples of variables, are special forms of a more general atom

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

saying that the variables $\overline{x}$ are completely independent from $\overline{y}$ for any constant value of $\overline{z}$.

Definition. A team X satisfies the atom $\overline{x} \perp_{\overline{z}} \overline{y}$ if for any pair of assignments $s, s' \in X$ with $s(\overline{z}) = s'(\overline{z})$ there is a third assignment $s'' \in X$ with

- $s''(\overline{z}) = s(\overline{z}) = s'(\overline{z})$
- $s''(\overline{x}) = s(\overline{x})$
- $s''(\overline{y}) = s'(\overline{y})$.

Dependence versus independence

How do dependence atoms and independence atoms compare with respect to expressive power?

Dependence versus independence

How do dependence atoms and independence atoms compare with respect to expressive power?

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

Dependence versus independence

How do dependence atoms and independence atoms compare with respect to expressive power?

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

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

Dependence versus independence

How do dependence atoms and independence atoms compare with respect to expressive power?

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

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

Corollary $=(\overline{z}, \overline{x}) \iff \overline{x} \perp_{\overline{z}} \overline{x}$

Dependence versus independence

How do dependence atoms and independence atoms compare with respect to expressive power?

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

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

Corollary $=(\overline{z}, \overline{x}) \iff \overline{x} \perp_{\overline{z}} \overline{x}$

So dependence is a special case of (the general form of) independence

Inclusion, exclusion, and all that

Besides dependence and independence, there are other interesting atomic properties of teams. One source of such properties is database dependency theory.

Inclusion, exclusion, and all that

Besides dependence and independence, there are other interesting atomic properties of teams. One source of such properties is database dependency theory.

Inclusion dependencies:

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

Inclusion, exclusion, and all that

Besides dependence and independence, there are other interesting atomic properties of teams. One source of such properties is database dependency theory.

Inclusion dependencies:

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

Exclusion dependencies:

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

Inclusion, exclusion, and all that

Besides dependence and independence, there are other interesting atomic properties of teams. One source of such properties is database dependency theory.

Inclusion dependencies:

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

Exclusion dependencies:

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

Equiextension:

$$\vDash_X (\overline{x} \bowtie \overline{y}) \quad :\Longleftrightarrow \quad \{s(\overline{x}) : s \in X\} = \{s(\overline{y}) : s \in X\}$$

Logics of dependence and independence

Combine the atoms stating dependencies and/or independencies with the common logical operators, such as 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 $\overline{x} \perp_{\overline{z}} \overline{y}$

Inclusion logic: FO + inclusion atoms $\overline{x} \subseteq \overline{y}$

and so on.

Logics of dependence and independence

Combine the atoms stating dependencies and/or independencies with the common logical operators, such as 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 $\overline{x} \perp_{\overline{z}} \overline{y}$

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, $\mathfrak{A} \models_X \psi(\overline{x})$?

Team semantics for first-order logic

$$\mathfrak{A} \models_X \psi(\overline{y}) \quad \Longleftrightarrow \quad \mathfrak{A} \models_s \psi(\overline{y}) \text{ for all } s \in X$$

Team semantics for first-order logic

$$\mathfrak{A} \models_X \psi(\overline{y}) \quad \Longleftrightarrow \quad \mathfrak{A} \models_s \psi(\overline{y}) \text{ for all } s \in X$$

- $\mathfrak{A} \models_X \psi \wedge \varphi \quad :\Longleftrightarrow \quad$ for all $s \in X$, $\mathfrak{A} \models_s \psi$ and $\mathfrak{A} \models_s \varphi$
- $\mathfrak{A} \models_X \psi \vee \varphi \quad :\Longleftrightarrow \quad$ for all $s \in X$, $\mathfrak{A} \models_s \psi$ or $\mathfrak{A} \models_s \varphi$
- $\mathfrak{A} \models_X \exists y \psi \quad :\Longleftrightarrow \quad$ for all $s \in X$ there exist $a \in A$ with $\mathfrak{A} \models_{s[y \mapsto a]} \psi$
- $\mathfrak{A} \models_X \forall y \psi \quad :\Longleftrightarrow \quad$ for all $s \in X$ and all $a \in A$, $\mathfrak{A} \models_{s[y \mapsto a]} \psi$

Team semantics: inductive definition

- For $\psi(\overline{y}) \in \mathrm{FO}$: $\quad \mathfrak{A} \models_X \psi(\overline{y}) \quad \Longleftrightarrow \quad \mathfrak{A} \models_s \psi(\overline{y})$ for all $s \in X$
- $\mathfrak{A} \models_X \psi \wedge \varphi \quad :\Longleftrightarrow \quad$ for all $s \in X$, $\mathfrak{A} \models_s \psi$ and $\mathfrak{A} \models_s \varphi$
- $\mathfrak{A} \models_X \psi \vee \varphi \quad :\Longleftrightarrow \quad$ for all $s \in X$, $\mathfrak{A} \models_s \psi$ or $\mathfrak{A} \models_s \varphi$
- $\mathfrak{A} \models_X \exists y \psi \quad :\Longleftrightarrow \quad$ for all $s \in X$ there exist $a \in A$ with $\mathfrak{A} \models_{s[y \mapsto a]} \psi$
- $\mathfrak{A} \models_X \forall y \psi \quad :\Longleftrightarrow \quad$ for all $s \in X$ and all $a \in A$, $\mathfrak{A} \models_{s[y \mapsto a]} \psi$

Team semantics: inductive definition

- For $\psi(\overline{y}) \in \mathrm{FO}$: $\quad \mathfrak{A} \models_X \psi(\overline{y}) \quad \Longleftrightarrow \quad \mathfrak{A} \models_s \psi(\overline{y})$ for all $s \in X$
- $\mathfrak{A} \models_X \psi \wedge \varphi \quad :\Longleftrightarrow \quad \mathfrak{A} \models_X \psi$ and $\mathfrak{A} \models_X \varphi$
- $\mathfrak{A} \models_X \psi \vee \varphi \quad :\Longleftrightarrow \quad$ for all $s \in X$, $\mathfrak{A} \models_s \psi$ or $\mathfrak{A} \models_s \varphi$
- $\mathfrak{A} \models_X \exists y \psi \quad :\Longleftrightarrow \quad$ for all $s \in X$ there exist $a \in A$ with $\mathfrak{A} \models_{s[y \mapsto a]} \psi$
- $\mathfrak{A} \models_X \forall y \psi \quad :\Longleftrightarrow \quad$ for all $s \in X$ and all $a \in A$, $\mathfrak{A} \models_{s[y \mapsto a]} \psi$

Team semantics: inductive definition

- For $\psi(\overline{y}) \in \mathrm{FO}$: $\quad \mathfrak{A} \models_X \psi(\overline{y}) \quad \Longleftrightarrow \quad \mathfrak{A} \models_s \psi(\overline{y})$ for all $s \in X$
- $\mathfrak{A} \models_X \psi \wedge \varphi \quad :\Longleftrightarrow \quad \mathfrak{A} \models_X \psi$ and $\mathfrak{A} \models_X \varphi$
- $\mathfrak{A} \models_X \psi \vee \varphi \quad :\Longleftrightarrow \quad X = Y \cup Z$ such that $\mathfrak{A} \models_Y \psi$ and $\mathfrak{A} \models_Z \varphi$
- $\mathfrak{A} \models_X \exists y \psi \quad :\Longleftrightarrow \quad$ for all $s \in X$ there exist $a \in A$ with $\mathfrak{A} \models_{s[y \mapsto a]} \psi$
- $\mathfrak{A} \models_X \forall y \psi \quad :\Longleftrightarrow \quad$ for all $s \in X$ and all $a \in A$, $\mathfrak{A} \models_{s[y \mapsto a]} \psi$

Team semantics: inductive definition

- For $\psi(\overline{y}) \in \mathrm{FO}$: $\quad \mathfrak{A} \models_X \psi(\overline{y}) \quad \Longleftrightarrow \quad \mathfrak{A} \models_s \psi(\overline{y})$ for all $s \in X$
- $\mathfrak{A} \models_X \psi \wedge \varphi \quad :\Longleftrightarrow \quad \mathfrak{A} \models_X \psi$ and $\mathfrak{A} \models_X \varphi$
- $\mathfrak{A} \models_X \psi \vee \varphi \quad :\Longleftrightarrow \quad X = Y \cup Z$ such that $\mathfrak{A} \models_Y \psi$ and $\mathfrak{A} \models_Z \varphi$
- $\mathfrak{A} \models_X \exists y \psi \quad :\Longleftrightarrow \quad$ for all $s \in X$ there exist $a \in A$ with $\mathfrak{A} \models_{s[y \mapsto a]} \psi$
- $\mathfrak{A} \models_X \forall y \psi \quad :\Longleftrightarrow \quad$ for all $s \in X$ and all $a \in A$, $\mathfrak{A} \models_{s[y \mapsto a]} \psi$

Notice that $(\varphi \vee \varphi)$ is, in general, not equivalent to φ

Team semantics: inductive definition

- For $\psi(\overline{y}) \in \mathrm{FO}$: $\quad \mathfrak{A} \models_X \psi(\overline{y}) \quad \Longleftrightarrow \quad \mathfrak{A} \models_s \psi(\overline{y})$ for all $s \in X$
- $\mathfrak{A} \models_X \psi \wedge \varphi \quad :\Longleftrightarrow \quad \mathfrak{A} \models_X \psi$ and $\mathfrak{A} \models_X \varphi$
- $\mathfrak{A} \models_X \psi \vee \varphi \quad :\Longleftrightarrow \quad X = Y \cup Z$ such that $\mathfrak{A} \models_Y \psi$ and $\mathfrak{A} \models_Z \varphi$
- $\mathfrak{A} \models_X \exists y \psi \quad :\Longleftrightarrow \quad$ for all $s \in X$ there exist $a \in A$ with $\mathfrak{A} \models_{s[y \mapsto a]} \psi$
- $\mathfrak{A} \models_X \forall y \psi \quad :\Longleftrightarrow \quad$ for all $s \in X$ and all $a \in A$, $\mathfrak{A} \models_{s[y \mapsto a]} \psi$

Notice that $(\varphi \vee \varphi)$ is, in general, not equivalent to φ

An example from dependence logic:

$=(y)$ means that the value of y is constant in the given team

$=(y) \vee =(y)$ means that y takes at most two values in the given team

Team semantics: inductive definition

- For $\psi(\overline{y}) \in \mathrm{FO}$: $\quad \mathfrak{A} \models_X \psi(\overline{y}) \quad \Longleftrightarrow \quad \mathfrak{A} \models_s \psi(\overline{y})$ for all $s \in X$
- $\mathfrak{A} \models_X \psi \wedge \varphi \quad :\Longleftrightarrow \quad \mathfrak{A} \models_X \psi$ and $\mathfrak{A} \models_X \varphi$
- $\mathfrak{A} \models_X \psi \vee \varphi \quad :\Longleftrightarrow \quad X = Y \cup Z$ such that $\mathfrak{A} \models_Y \psi$ and $\mathfrak{A} \models_Z \varphi$
- $\mathfrak{A} \models_X \exists y \psi \quad :\Longleftrightarrow \quad \exists F : X \to \mathcal{P}(A) \setminus \{\varnothing\}$ such that $\mathfrak{A} \models_{X[y \mapsto F]} \psi$
- $\mathfrak{A} \models_X \forall y \psi \quad :\Longleftrightarrow \quad$ for all $s \in X$ and all $a \in A$, $\mathfrak{A} \models_{s[y \mapsto a]} \psi$

Team semantics: inductive definition

- For $\psi(\overline{y}) \in \mathrm{FO}$: $\quad \mathfrak{A} \models_X \psi(\overline{y}) \quad \Longleftrightarrow \quad \mathfrak{A} \models_s \psi(\overline{y})$ for all $s \in X$
- $\mathfrak{A} \models_X \psi \wedge \varphi \quad :\Longleftrightarrow \quad \mathfrak{A} \models_X \psi$ and $\mathfrak{A} \models_X \varphi$
- $\mathfrak{A} \models_X \psi \vee \varphi \quad :\Longleftrightarrow \quad X = Y \cup Z$ such that $\mathfrak{A} \models_Y \psi$ and $\mathfrak{A} \models_Z \varphi$
- $\mathfrak{A} \models_X \exists y \psi \quad :\Longleftrightarrow \quad \exists F : X \to \mathcal{P}(A) \setminus \{\varnothing\}$ such that $\mathfrak{A} \models_{X[y \mapsto F]} \psi$
- $\mathfrak{A} \models_X \forall y \psi \quad :\Longleftrightarrow \quad$ for all $s \in X$ and all $a \in A$, $\mathfrak{A} \models_{s[y \mapsto a]} \psi$

Choose (for every $s \in X$) an arbitrary non-empty set of witnesses for $\exists x \ldots$
rather than just a single witness: lax semantics as opposed to strict semantics.

For FO and dependence logic the difference is immaterial
For stronger logics, only lax semantics guarantees the locality principle:

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

Team semantics: inductive definition

- For $\psi(\overline{y}) \in \mathrm{FO}$: $\quad \mathfrak{A} \models_X \psi(\overline{y}) \iff \mathfrak{A} \models_s \psi(\overline{y})$ for all $s \in X$
- $\mathfrak{A} \models_X \psi \wedge \varphi \quad :\iff \quad \mathfrak{A} \models_X \psi$ and $\mathfrak{A} \models_X \varphi$
- $\mathfrak{A} \models_X \psi \vee \varphi \quad :\iff \quad X = Y \cup Z$ such that $\mathfrak{A} \models_Y \psi$ and $\mathfrak{A} \models_Z \varphi$
- $\mathfrak{A} \models_X \exists y \psi \quad :\iff \quad \exists F : X \to \mathcal{P}(A) \setminus \{\emptyset\}$ such that $\mathfrak{A} \models_{X[y \mapsto F]} \psi$
- $\mathfrak{A} \models_X \forall y \psi \quad :\iff \quad \mathfrak{A} \models_{X[y \mapsto A]} \psi$

Team semantics: inductive definition

- For $\psi(\overline{y}) \in \mathrm{FO}$: $\quad \mathfrak{A} \models_X \psi(\overline{y}) \quad \Longleftrightarrow \quad \mathfrak{A} \models_s \psi(\overline{y})$ for all $s \in X$
- $\mathfrak{A} \models_X \psi \wedge \varphi \quad :\Longleftrightarrow \quad \mathfrak{A} \models_X \psi$ and $\mathfrak{A} \models_X \varphi$
- $\mathfrak{A} \models_X \psi \vee \varphi \quad :\Longleftrightarrow \quad X = Y \cup Z$ such that $\mathfrak{A} \models_Y \psi$ and $\mathfrak{A} \models_Z \varphi$
- $\mathfrak{A} \models_X \exists y \psi \quad :\Longleftrightarrow \quad \exists F : X \to \mathcal{P}(A) \setminus \{\varnothing\}$ such that $\mathfrak{A} \models_{X[y \mapsto F]} \psi$
- $\mathfrak{A} \models_X \forall y \psi \quad :\Longleftrightarrow \quad \mathfrak{A} \models_{X[y \mapsto A]} \psi$

For sentences we define: $\mathfrak{A} \models \psi \;:\Longleftrightarrow \mathfrak{A} \models_{\{\varnothing\}} \psi$

Notice that we cannot reasonably replace $\{\varnothing\}$ by $\varnothing$ since the empty team satisfies all formulae: $\mathfrak{A} \models_\varnothing \psi$ for all ψ

Example: Defining 3-SAT in dependence logic

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

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

Example: 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	-

Example: Defining 3-SAT in dependence logic

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

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

Example: 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. φ is satisfiable if, and only if, the team Z_φ is a model of

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

From team semantics to Tarski semantics

A team X of assignments $s : V \to A$ can be represented as a relation $\mathrm{rel}(X) \subseteq A^{|V|}$.

The translation to Tarski semantics requires that we go to existential second-order logic Σ^1_1.

Proposition. Every formula $\psi(x_1, \ldots, x_n)$ in dependence or independence logic, with vocabulary τ, can be translated into a Σ^1_1-sentence ψ^* of vocabulary $\tau \cup \{R\}$ such that

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

Indeed this holds for any extension of FO by atomic properties of teams that are first-order expressible (on the relation describing the team).

From team semantics to Tarski semantics

A team X of assignments $s : V \to A$ can be represented as a relation $\mathrm{rel}(X) \subseteq A^{|V|}$.

The translation to Tarski semantics requires that we go to existential second-order logic Σ^1_1.

Proposition. Every formula $\psi(x_1, \ldots, x_n)$ in dependence or independence logic, with vocabulary τ, can be translated into a Σ^1_1-sentence ψ^* of vocabulary $\tau \cup \{R\}$ such that

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

Indeed this holds for any extension of FO by atomic properties of teams that are first-order expressible (on the relation describing the team).

For sentences, dependence logic, and thus also independence logic coincides in expressive power with Σ^1_1.

The translation from dependence logic into Σ^1_1

Construct translation $\psi(x_1, \ldots, x_n) \quad \mapsto \quad \psi^*(R)$ such that

$$\mathfrak{A} \vDash_X \psi(\overline{x}) \quad \Longleftrightarrow \quad (\mathfrak{A}, \mathrm{rel}(X)) \vDash \psi^*$$

The translation from dependence logic into Σ^1_1

Construct translation $\psi(x_1, \ldots, x_n) \quad \mapsto \quad \psi^*(R)$ such that

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

- first-order literals $\alpha(\overline{x})$ are translated into $\forall \overline{x}(R\overline{x} \to \alpha(\overline{x}))$

The translation from dependence logic into Σ^1_1

Construct translation $\psi(x_1, \ldots, x_n) \quad \mapsto \quad \psi^*(R)$ such that

$$\mathfrak{A} \vDash_X \psi(\overline{x}) \quad \Longleftrightarrow \quad (\mathfrak{A}, \mathrm{rel}(X)) \vDash \psi^*$$

- first-order literals $\alpha(\overline{x})$ are translated into $\forall \overline{x}(R\overline{x} \to \alpha(\overline{x}))$
- dependency atoms $=(x_{i_1}, \ldots, x_{i_k}, x_i)$ are translated into

$$\forall \overline{x} \forall \overline{y} \Big(R\overline{x} \wedge R\overline{y} \wedge \bigwedge_{j=1}^{k} (x_{i_j} = y_{i_j}) \to (x_i = y_i) \Big)$$

The translation from dependence logic into Σ^1_1

Construct translation $\psi(x_1, \ldots, x_n) \quad \mapsto \quad \psi^*(R)$ such that

$$\mathfrak{A} \vDash_X \psi(\overline{x}) \quad \Longleftrightarrow \quad (\mathfrak{A}, \mathrm{rel}(X)) \vDash \psi^*$$

- first-order literals $\alpha(\overline{x})$ are translated into $\forall \overline{x}(R\overline{x} \to \alpha(\overline{x}))$
- dependency atoms $=(x_{i_1}, \ldots, x_{i_k}, x_i)$ are translated into

$$\forall \overline{x} \forall \overline{y} \Big(R\overline{x} \wedge R\overline{y} \wedge \bigwedge_{j=1}^{k} (x_{i_j} = y_{i_j}) \to (x_i = y_i) \Big)$$

- $(\psi(\overline{x}) \wedge \varphi(\overline{x}))^* := \psi^*(R) \wedge \varphi^*(R)$

The translation from dependence logic into Σ^1_1

Construct translation $\psi(x_1, \ldots, x_n) \quad \mapsto \quad \psi^*(R)$ such that

$$\mathfrak{A} \vDash_X \psi(\overline{x}) \quad \Longleftrightarrow \quad (\mathfrak{A}, \mathrm{rel}(X)) \vDash \psi^*$$

- first-order literals $\alpha(\overline{x})$ are translated into $\forall \overline{x}(R\overline{x} \to \alpha(\overline{x}))$
- dependency atoms $=(x_{i_1}, \ldots, x_{i_k}, x_i)$ are translated into

$$\forall \overline{x} \forall \overline{y} \Big(R\overline{x} \wedge R\overline{y} \wedge \bigwedge_{j=1}^{k} (x_{i_j} = y_{i_j}) \to (x_i = y_i) \Big)$$

- $(\psi(\overline{x}) \wedge \varphi(\overline{x}))^* := \psi^*(R) \wedge \varphi^*(R)$
- $(\psi(\overline{x}) \vee \varphi(\overline{x}))^* := \exists S \exists T (\forall \overline{x}(R\overline{x} \to (S\overline{x} \vee T\overline{x})) \wedge \psi^*(S) \wedge \varphi^*(T))$

The translation from dependence logic into Σ^1_1

Construct translation $\psi(x_1, \ldots, x_n) \quad \mapsto \quad \psi^*(R)$ such that

$$\mathfrak{A} \vDash_X \psi(\overline{x}) \quad \Longleftrightarrow \quad (\mathfrak{A}, \mathrm{rel}(X)) \vDash \psi^*$$

- first-order literals $\alpha(\overline{x})$ are translated into $\forall \overline{x}(R\overline{x} \to \alpha(\overline{x}))$
- dependency atoms $=(x_{i_1}, \ldots, x_{i_k}, x_i)$ are translated into

$$\forall \overline{x} \forall \overline{y} \Big(R\overline{x} \wedge R\overline{y} \wedge \bigwedge_{j=1}^{k} (x_{i_j} = y_{i_j}) \to (x_i = y_i) \Big)$$

- $(\psi(\overline{x}) \wedge \varphi(\overline{x}))^* := \psi^*(R) \wedge \varphi^*(R)$
- $(\psi(\overline{x}) \vee \varphi(\overline{x}))^* := \exists S \exists T (\forall \overline{x}(R\overline{x} \to (S\overline{x} \vee T\overline{x})) \wedge \psi^*(S) \wedge \varphi^*(T))$
- $(\forall y\, \psi)^* := \exists S (\forall \overline{x} \forall y (R\overline{x} \to S\overline{x}y) \wedge \psi^*(S))$

The translation from dependence logic into Σ^1_1

Construct translation $\psi(x_1, \ldots, x_n) \quad \mapsto \quad \psi^*(R)$ such that

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

- first-order literals $\alpha(\overline{x})$ are translated into $\forall \overline{x}(R\overline{x} \to \alpha(\overline{x}))$
- dependency atoms $=(x_{i_1}, \ldots, x_{i_k}, x_i)$ are translated into

$$\forall \overline{x} \forall \overline{y} \Big(R\overline{x} \wedge R\overline{y} \wedge \bigwedge_{j=1}^{k} (x_{i_j} = y_{i_j}) \to (x_i = y_i) \Big)$$

- $(\psi(\overline{x}) \wedge \varphi(\overline{x}))^* := \psi^*(R) \wedge \varphi^*(R)$
- $(\psi(\overline{x}) \vee \varphi(\overline{x}))^* := \exists S \exists T (\forall \overline{x}(R\overline{x} \to (S\overline{x} \vee T\overline{x})) \wedge \psi^*(S) \wedge \varphi^*(T))$
- $(\forall y\, \psi)^* := \exists S (\forall \overline{x} \forall y (R\overline{x} \to S\overline{x}y) \wedge \psi^*(S))$
- $(\exists y\, \psi)^* := \exists S (\forall \overline{x} \exists y (R\overline{x} \to S\overline{x}y) \wedge \psi^*(S))$

Expressive power of dependence logic: formulae

Downwards Closure: If $\mathfrak{A} \models_X \psi$ and $Y \subseteq X$, then $\mathfrak{A} \models_Y \psi$

Hence, the sentences ψ^* have to be downwards monotone:
If $(\mathfrak{A}, R) \models \psi^*$ and $S \subseteq R$ then $(\mathfrak{A}, S) \models \psi^*$.

Expressive power of dependence logic: formulae

Downwards Closure: If $\mathfrak{A} \models_X \psi$ and $Y \subseteq X$, then $\mathfrak{A} \models_Y \psi$

Hence, the sentences ψ^* have to be downwards monotone:
If $(\mathfrak{A}, R) \models \psi^*$ and $S \subseteq R$ then $(\mathfrak{A}, S) \models \psi^*$.

Thus dependence logic is a strict fragment of existential second-order logic.

Expressive power of dependence logic: formulae

Downwards Closure: If $\mathfrak{A} \models_X \psi$ and $Y \subseteq X$, then $\mathfrak{A} \models_Y \psi$

Hence, the sentences ψ^* have to be downwards monotone:
If $(\mathfrak{A}, R) \models \psi^*$ and $S \subseteq R$ then $(\mathfrak{A}, S) \models \psi^*$.

Thus dependence logic is a strict fragment of existential second-order logic.

Theorem (Kontinen and Väänänen 2009)
The expressive power of formulae $\psi(x_1, \ldots, x_n)$ of dependence logic is precisely that of existential second-order sentences with the predicate for the team occurring only negatively.

Expressive power of dependence logic: formulae

Downwards Closure: If $\mathfrak{A} \models_X \psi$ and $Y \subseteq X$, then $\mathfrak{A} \models_Y \psi$

Hence, the sentences ψ^* have to be downwards monotone:
If $(\mathfrak{A}, R) \models \psi^*$ and $S \subseteq R$ then $(\mathfrak{A}, S) \models \psi^*$.

Thus dependence logic is a strict fragment of existential second-order logic.

Theorem (Kontinen and Väänänen 2009)
The expressive power of formulae $\psi(x_1, \ldots, x_n)$ of dependence logic is precisely that of existential second-order sentences with the predicate for the team occurring only negatively.

While sentences of dependence logic can express all NP-properties of finite structures, only the downwards closed NP-properties of teams are definable by open formulae of dependence logic.

The translation from independence logic into Σ^1_1

Construct translation $\psi(x_1, \ldots, x_n) \quad \mapsto \quad \psi^*(R)$ such that

$$\mathfrak{A} \vDash_X \psi(\overline{x}) \quad \Longleftrightarrow \quad (\mathfrak{A}, \mathrm{rel}(X)) \vDash \psi^*$$

- first-order literals $\alpha(\overline{x})$ are translated into $\forall \overline{x}(R\overline{x} \to \alpha(\overline{x}))$
- independence atoms $x_i \perp_{x_k} x_j$ are translated into

$$\forall \overline{x} \forall \overline{y} \Big(R\overline{x} \wedge R\overline{y} \wedge (x_k = y_k) \to \exists \overline{z} \Big(R\overline{z} \wedge (z_k = x_k) \wedge (z_i = x_i) \wedge (z_j = y_j) \Big) \Big)$$

- $(\psi(\overline{x}) \wedge \varphi(\overline{x}))^* := \psi^*(R) \wedge \varphi^*(R)$
- $(\psi(\overline{x}) \vee \varphi(\overline{x}))^* := \exists S \exists T (\forall \overline{x}(R\overline{x} \leftrightarrow (S\overline{x} \vee T\overline{x})) \wedge \psi^*(S) \wedge \varphi^*(T))$
- $(\forall y\, \psi)^* := \exists S (\forall \overline{x} \forall y (R\overline{x} \leftrightarrow S\overline{x}y) \wedge \psi^*(S))$
- $(\exists y\, \psi)^* := \exists S (\forall \overline{x} \exists y (R\overline{x} \leftrightarrow S\overline{x}y) \wedge \psi^*(S))$

Expressive power of independence logic

Contrary to dependence logic, inclusion logic and independence logic are not downwards closed.

For instance $x \perp y$ is not preserved under subteams.

Inclusion logic is closed under union. Independence logic is not.

Expressive power of independence logic

Contrary to dependence logic, inclusion logic and independence logic are not downwards closed.

For instance $x \perp y$ is not preserved under subteams.

Inclusion logic is closed under union. Independence logic is not.

- Exclusion logic $\equiv$ dependence logic.
- Inclusion logic and dependence logic are incomparable.
- FO + inclusion + exclusion $\equiv$ independence logic.
- Independence logic coincides with Σ^1_1 also wrt the expressive power of open formulae.

Corollary. All NP-properties of teams are definable in independence logic.

Negation

We have permitted negation only applied to first-order atoms. But we can define an operator $\psi \mapsto \psi^{\neg}$, taking the negation normal form of $\neg\psi$ (with the convention that negated dependency atoms are true only in the empty team)

Negation

We have permitted negation only applied to first-order atoms. But we can define an operator $\psi \mapsto \psi^{\neg}$, taking the negation normal form of $\neg\psi$ (with the convention that negated dependency atoms are true only in the empty team)

Negation is not a semantic operation, contrary to disjunction, conjunction, and quantifiers.

Negation

We have permitted negation only applied to first-order atoms. But we can define an operator $\psi \mapsto \psi^{\neg}$, taking the negation normal form of $\neg\psi$ (with the convention that negated dependency atoms are true only in the empty team)

Negation is not a semantic operation, contrary to disjunction, conjunction, and quantifiers.

Define the semantic extension of ψ on $\mathfrak{A}$ as $[\![\psi]\!]^{\mathfrak{A}} := \{X : \mathfrak{A} \vDash_X \psi\}$.

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 for quantifiers.

However, knowing $[\![\psi]\!]^{\mathfrak{A}}$ does not provide much knowledge about $[\![\psi^{\neg}]\!]^{\mathfrak{A}}$.

Negation and interpolation

We only consider structures with at least wo elements. Two formulae ψ and φ are contradictory if $[\![\psi]\!]^{\mathfrak{A}} \cap [\![\varphi]\!]^{\mathfrak{A}} = \{\varnothing\}$ for any structure $\mathfrak{A}$.

Proposition. (Kontinen and Väänänen) For any two contradictory formulae ψ and φ of dependence logic there is a formula ϑ such that $[\![\vartheta]\!]^{\mathfrak{A}} = [\![\psi]\!]^{\mathfrak{A}}$ and $[\![\vartheta^{\neg}]\!]^{\mathfrak{A}} = [\![\varphi]\!]^{\mathfrak{A}}$ for all $\mathfrak{A}$.

Negation and interpolation

We only consider structures with at least wo elements. Two formulae ψ and φ are contradictory if $[\![\psi]\!]^{\mathfrak{A}} \cap [\![\varphi]\!]^{\mathfrak{A}} = \{\varnothing\}$ for any structure $\mathfrak{A}$.

Proposition. (Kontinen and Väänänen) For any two contradictory formulae ψ and φ of dependence logic there is a formula ϑ such that $[\![\vartheta]\!]^{\mathfrak{A}} = [\![\psi]\!]^{\mathfrak{A}}$ and $[\![\vartheta^{\neg}]\!]^{\mathfrak{A}} = [\![\varphi]\!]^{\mathfrak{A}}$ for all $\mathfrak{A}$.

In this form, this is not true in general for logics with team semantics. For instance independence logic contains $\forall x(x \subseteq y)$ and $=(y)$ which are contradictory but we shall see that they have no such interpolant.

Strongly contradictory formulae

Two formulae ψ and φ are strongly contradictory if $X \cap Y = \varnothing$ for all teams X, Y and all structures $\mathfrak{A}$ such that $\mathfrak{A} \models_X \psi$ and $\mathfrak{A} \models_Y \varphi$.

Lemma. Every formula is strongly contradictory to its negation.

Corollary. Only a pair of strongly contradictory formulae ψ, φ can have an interpolant ϑ with $\psi \equiv \vartheta$ and $\varphi \equiv \vartheta^{\neg}$.

The formulae $\forall x(x \subseteq y)$ and $=(y)$ are contradictory (on structures with at least two elements) but have no interpolant, since they are not strongly contradictory. Indeed, let s_a be the assignment $y \mapsto a$. Then $X = \{s_a : a \in A\} \in [\![\forall x(x \subseteq y)]\!]^{\mathfrak{A}}$. Further, $\{s_a\} \in [\![=(y)]\!]^{\mathfrak{A}}$ for all $a \in A$. However, $X \cap \{s_a\} = \{s_a\} \neq \varnothing$.

Being strongly contradictory is a necessary condition for having an interpolant. It turns out that it is also sufficient.

Completely undetermined sentences

A sentence is completely undetermined if neither the sentence itself, nor its negation is true on any structure with more than one element.

Examples: $\forall x =(x)$ or $\forall x \exists y (x \perp y \wedge x = y)$

Proposition. Let L be any logic with team semantics, closed under first-order operations, containing a completely undetermined sentence $\perp_\perp$. Then for any $\psi, \varphi \in L$, the following are equivalent

(1) There is a formula η such that $\psi \models \eta$ and $\varphi \models \eta^\neg$

(2) There is a formula ϑ such that $\psi \equiv \vartheta$ and $\varphi \equiv \vartheta^\neg$

Proof. (2) $\Rightarrow$ (1): Take $\eta = \vartheta$.

(1) $\Rightarrow$ (2): Take $\vartheta := (\psi \vee \perp_\perp) \wedge ((\varphi \vee \perp_\perp)^\neg \vee \eta)$. Then

$\vartheta \equiv \psi \wedge (\perp \vee \eta) \equiv \psi \wedge \eta \equiv \psi$

$\vartheta^\neg = (\psi \vee \perp_\perp)^\neg \vee ((\varphi \vee \perp_\perp) \wedge \eta^\neg) \equiv \perp \vee (\varphi \wedge \eta^\neg) \equiv \varphi \wedge \eta^\neg \equiv \varphi.$

1-closed formulae

A formula ψ 1-closed if whenever $\mathfrak{A} \models_X \psi$ then also $\mathfrak{A} \models_{\{s\}} \psi$ for all $s \in X$.

All formulae of dependence logic are 1-closed. Further independence atoms (and in fact all purely existential formulae of independence logic) are 1-closed.

Lemma. Let L be a logic with team semantics, that is closed under first-order operations and translatable into Σ^1_1. Then for every formula $\psi \in L$ there exists a formula $\psi^{\downarrow}$ in dependence logic such that the teams satisfying $\psi^{\downarrow}$ are exactly the subteams of the teams satisfying ψ.

Proof. Translate ψ into $\psi^*(Y) \in \Sigma^1_1$ and let $\varphi(X) = \exists Y(\forall \overline{x}(X\overline{x} \to Y\overline{x}) \wedge \psi^*(Y))$. Since X appears only negatively in $\varphi(X)$, it can be translated into an equivalent formula $\psi^{\downarrow}$.

Notice that $\psi \models \psi^{\downarrow}$ and that $\psi^{\downarrow}$ is 1-closed. Further ψ and φ are strongly contradictory if, and only if, $\psi^{\downarrow}$ and $\varphi^{\downarrow}$ are contradictory (and hence strongly contradictory).

The Interpolation Theorem

Let L be a logic with team semantics, which contains FO and can be embedded into Σ^1_1, and which has a totally undetermined sentence.

Theorem. For any two strongly contradictory formulae ψ, φ from L, there exists a formula ϑ in L such that $\psi \equiv \vartheta$ and $\varphi \equiv \vartheta^{\neg}$.

Let $\exists \overline{R} \tilde{\psi}(\overline{R}, X)$ and $\exists \overline{S} \tilde{\varphi}(\overline{S}, X)$ be Σ^1_1-translations of $\psi^{\downarrow}$ and $\varphi^{\downarrow}$. Then $\tilde{\psi}(\overline{R}, X) \models (\neg \tilde{\varphi}(\overline{S}, X) \vee X = \varnothing)$ is a valid first-order implication.

By Craig's Interpolation Theorem there is a first-order sentence $\tilde{\eta}(X)$ such that $\tilde{\psi}(\overline{R}, X) \models \tilde{\eta}(X)$ and $(\tilde{\varphi}(\overline{S}, X) \wedge X \neq \varnothing) \models \neg \tilde{\eta}(X)$.

Let $\eta(\overline{x}) := \tilde{\eta}[X\overline{z}/\overline{z} = \overline{x}]$. For any team $\{s\}$ of size one, we have $(\mathfrak{A}, \{s\}) \models \tilde{\eta}$ if, and only if, $\mathfrak{A} \models_{\{s\}} \eta$.

Then $\psi \models \eta$ and $\varphi \models \neg\eta$ (and hence an interpolant ϑ exists)

Part II: Model-Checking Games

- Model-checking games for first-order logic
- Second-order reachability games
- Model-checking games for logics with team semantics
- Examples
- Complexity
- Deterministic versus nondeterministic strategies

Model-Checking Games

The model checking problem for a logic L (with classical Tarski-semantics)

Given: structure $\mathfrak{A}$

formula $\psi(\overline{x}) \in L$

assignment $s : \text{free}(\psi) \to A$

Question: $\mathfrak{A} \models_s \psi$?

Model-Checking Games

The model checking problem for a logic L (with classical Tarski-semantics)

Given: structure $\mathfrak{A}$
formula $\psi(\overline{x}) \in L$
assignment $s : \text{free}(\psi) \to A$

Question: $\mathfrak{A} \models_s \psi$?

Reduce model checking problem $\mathfrak{A} \models_s \psi$ to strategy problem for model checking game $G(\mathfrak{A}, \psi, s)$, played by

- Falsifier (also called Player 1), and
- Verifier (also called Player 0), such that

$$\mathfrak{A} \models_s \psi \iff \text{Verifier has winning strategy for } G(\mathfrak{A}, \psi, s)$$

Model-Checking Games

The model checking problem for a logic L (with classical Tarski-semantics)

Given: structure $\mathfrak{A}$
formula $\psi(\overline{x}) \in L$
assignment $s : \text{free}(\psi) \to A$

Question: $\mathfrak{A} \models_s \psi$?

Reduce model checking problem $\mathfrak{A} \models_s \psi$ to strategy problem for model checking game $G(\mathfrak{A}, \psi, s)$, played by

- Falsifier (also called Player 1), and
- Verifier (also called Player 0), such that

$$\mathfrak{A} \models_s \psi \iff \text{Verifier has winning strategy for } G(\mathfrak{A}, \psi, s)$$

$\Longrightarrow$ Model checking via construction of winning strategies

Model-checking game for first-order logic

The game $\mathcal{G}(\mathfrak{A}, \psi, t)$ for a structure $\mathfrak{A}$ and $\psi(\overline{x}) \in \mathrm{FO}$.

Positions: (φ, s) φ is a subformula of ψ and $s : \mathrm{free}(\varphi) \to A$

Verifier moves:

$(\varphi_1 \vee \varphi_2, s) \to (\varphi_i, s \restriction \mathrm{free}(\varphi_i))$ $(i = 1, 2)$

$(\exists x \varphi, s) \to (\varphi, s[x \mapsto a])$ $(a \in A)$

Falsifier moves

$(\varphi_1 \wedge \varphi_2, s) \to (\varphi_i, s \restriction \mathrm{free}(\varphi_i))$ $(i = 1, 2)$

$(\forall x \varphi, s) \to (\varphi, s[x \mapsto a])$ $(a \in A)$

Terminal positions: φ atomic / negated atomic

$$\begin{matrix}\text{Verifier}\\ \text{Falsifier}\end{matrix} \text{ wins at } (\varphi, s) \iff \mathfrak{A} \begin{matrix}\models_s\\ \not\models_s\end{matrix} \varphi$$

Reachability games

Game graph $\mathcal{G} = (V, V_0, V_1, T, I, E)$, $\quad V = V_0 \cup V_1 \cup T$, $\quad E \subseteq V \times V$

- Player 0 moves from positions $v \in V_0$,
  Player 1 moves from $v \in V_1$,
  T is the set of terminal positions.
- I is the set of initial positions
- Moves are along edges.
  Hence plays are finite or infinite paths through the graph
- Winning condition $\mathrm{Win} \subseteq T$: at a terminal position $v \in T$,
  Player 0 has won if $v \in \mathrm{Win}$ and Player 1 has won if $v \in T \setminus \mathrm{Win}$.
  Infinite plays are draws.

Winning strategies

A (nondeterministic) strategy of Player 0 for $\mathcal{G}$, defined on $W \subseteq V$, is a subgraph $S = (W, F) \subseteq (V, E)$ such that

(1) if $v \in V_0 \cap W$ then $vF \neq \emptyset$,

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

The strategy $S = (W, F)$ is winning from $X \subseteq I$ if $X \subseteq W$ and all plays that start at some $v \in X$ and are consistent with S reach a winning terminal position $w \in \mathrm{Win}$.

Given a reachability game game $(\mathcal{G}, \mathrm{Win})$ on a finite game graph, one can compute in linear time $O(|V| + |E|)$

- the maximal subset $X \subseteq I$ from which Player 0 can win, and
- a winning strategy from X.

Reachability games for logics with team semantics

Reduce model checking problem $\mathfrak{A} \models_X \psi$ to a reachability game $G(\mathfrak{A}, X, \psi)$.

Positions: (φ, Y) Y is now a team of assignments $s : \text{free}(\varphi) \to A$

Reachability games for logics with team semantics

Reduce model checking problem $\mathfrak{A} \models_X \psi$ to a reachability game $G(\mathfrak{A}, X, \psi)$.

Positions: (φ, Y) Y is now a team of assignments $s : \mathrm{free}(\varphi) \to A$

Falsifier moves:

$(\varphi_1 \wedge \varphi_2, Y) \to (\varphi_i, Y \upharpoonright \mathrm{free}(\varphi_i))$ $(i = 1, 2)$

$(\forall x \varphi, Y) \to (\varphi, Y[x \mapsto A])$

Verifier moves:

$(\exists x \varphi, Y) \to (\varphi, Y[x \mapsto F])$ for some $F : Y \to (\mathcal{P}(A) \setminus \{\varnothing\})$

Protocol for disjunctions:

At $(\varphi_1 \vee \varphi_2, Y)$ Verifier selects Y_1, Y_2 such that $Y_1 \cup Y_2 = Y$.

Falsifier selects $i \in \{1, 2\}$ and moves to $(\varphi_i, Y_i \upharpoonright \mathrm{free}(\varphi_i))$

Terminal positions: φ atomic / negated atomic

Do reachability games provide a satisfactory game-theoretic analysis for logics with team semantics?

The constructed reachability game really is the model-checking game for the translated formula into existential second-order logic.

Do reachability games provide a satisfactory game-theoretic analysis for logics with team semantics?

The constructed reachability game really is the model-checking game for the translated formula into existential second-order logic.

The size of this model is exponentially larger than the first-order game.

Do reachability games provide a satisfactory game-theoretic analysis for logics with team semantics?

The constructed reachability game really is the model-checking game for the translated formula into existential second-order logic.

The size of this model is exponentially larger than the first-order game.

It is not clear how to extract complexity results for model-checking problems out of these games

Do reachability games provide a satisfactory game-theoretic analysis for logics with team semantics?

The constructed reachability game really is the model-checking game for the translated formula into existential second-order logic.

The size of this model is exponentially larger than the first-order game.

It is not clear how to extract complexity results for model-checking problems out of these games

The game for the negated formula is not obtained by a simple dualization operation or a role switch of players

Second-order reachability games

Game graph: $\mathcal{G} = (V, V_0, V_1, T, I, E)$

Winning condition for Player 0: Win $\subseteq \mathcal{P}(T)$.

Does Player 0 have a (nondeterministic) strategy such that the set of terminal positions that are reachable by a play from I that is consistent with the strategy belongs to Win ?

A consistent winning strategy of Player 0 from X is a strategy $S = (W, F) \subseteq (V, E)$ such that

(1) W is the set of nodes that are reachable from X via edges in F

(2) $W \cap T \in$ Win.

Complexity

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

Complexity

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

The problem is obviously in NP: guess a subgraph, and verify that it is a consistent winning strategy.

Complexity

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

The problem is obviously in NP: guess a subgraph, and verify that it is a consistent winning strategy.

NP-hardness by reduction from SAT. Given a CNF-formula ψ consider the obvious game $G(\psi)$, where Player 1 can move from the initial position ψ to clauses, and Player 0 from clauses to their literals, with

$T := \{(C, Y) : C \text{ is a clause of } \psi, Y \in C\}$

$\text{Win} := \{U \subseteq T : U \text{ contains no conflicting pair } (C, Y), (C', \overline{Y})\}$

Player 0 has a consistent winning strategy for $G(\psi)$ $\iff$ ψ is satisfiable

Model-checking game for logics with team semantics

Consider FO together with a collection of atomic properties of teams.

Appropriate model checking games are obtained as follows:

- Take the same game graphs $\mathcal{G}(\mathfrak{A}, \psi)$ as for FO with Tarski-semantics (making sure that distinct occurrences of the same subformula are represented by distinct nodes).
- The set of initial positions is $I := \{(\psi, s) : s : \mathrm{free}(\psi) \to A\}$.
- The winning condition is a second-order reachability condition.
- This imposes consistency conditions on the admissible strategies.

Model-checking game for logics with team semantics

Consider FO together with a collection of atomic properties of teams.

Appropriate model checking games are obtained as follows:

- Take the same game graphs $\mathcal{G}(\mathfrak{A}, \psi)$ as for FO with Tarski-semantics (making sure that distinct occurrences of the same subformula are represented by distinct nodes).
- The set of initial positions is $I := \{(\psi, s) : s : \mathrm{free}(\psi) \to A\}$.
- The winning condition is a second-order reachability condition.
- This imposes consistency conditions on the admissible strategies.

The resulting games $\mathcal{G}(\mathfrak{A}, \psi)$, where Verifier may use only consistent strategies, can be viewed as games of imperfect information.

Second-order reachability games for model checking

Let (V, V_0, V_1, T, I, E) be the game graph of a model-checking game $\mathcal{G}(\mathfrak{A}, \psi)$.
Recall that $V = \{(\varphi, s) : \varphi \text{ is a subformula of } \psi,\ s : \text{free}(\varphi) \to A\}$

Second-order reachability games for model checking

Let (V, V_0, V_1, T, I, E) be the game graph of a model-checking game $\mathcal{G}(\mathfrak{A}, \psi)$.
Recall that $V = \{(\varphi, s) : \varphi \text{ is a subformula of } \psi,\ s : \text{free}(\varphi) \to A\}$

Teams defined by strategies: For any subset $U \subseteq V$ and any subformula φ of ψ

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

Second-order reachability games for model checking

Let (V, V_0, V_1, T, I, E) be the game graph of a model-checking game $\mathcal{G}(\mathfrak{A}, \psi)$.
Recall that $V = \{(\varphi, s) : \varphi \text{ is a subformula of } \psi,\ s : \text{free}(\varphi) \to A\}$

Teams defined by strategies: For any subset $U \subseteq V$ and any subformula φ of ψ

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

Second-order reachability condition $\text{Win} \subseteq \mathcal{P}(T)$:

$T = \{(\varphi, s) \in V : \varphi \text{ is an atomic or negated atomic subformula of } \psi\}$

$\text{Win} := \{U \subseteq T : \mathfrak{A} \models_{\text{Team}(U,\varphi)} \varphi \text{ for every atomic/negated atomic } \varphi\}$.

A consistent winning strategy for Player 0 (Verifier) $S = (W, F)$ thus has to guarantee that $\mathfrak{A} \models_{\text{Team}(W,\varphi)} \varphi$ for every atomic or negated atomic formula φ.

Notice that this refers to the entire set of terminal positions that are reachable by plays that are consistent with the strategy.

Correctness of the model checking games

The consistency condition for winning strategies translates from the atomic formulae to all positions of the game.

If $S = (W, F)$ is a consistent winning strategy for $\mathcal{G}(\mathfrak{A}, \psi)$ then, for all subformulae φ of ψ we have that $\mathfrak{A} \models_{\mathrm{Team}(S,\varphi)} \varphi$.

Correctness of the model checking games

The consistency condition for winning strategies translates from the atomic formulae to all positions of the game.

If $S = (W, F)$ is a consistent winning strategy for $\mathcal{G}(\mathfrak{A}, \psi)$ then, for all subformulae φ of ψ we have that $\mathfrak{A} \models_{\mathrm{Team}(S,\varphi)} \varphi$.

Theorem.

$\mathfrak{A} \models_X \psi \iff$ Verifier has a consistent winning strategy S for $\mathcal{G}(\mathfrak{A}, \psi)$ with $\mathrm{Team}(S, \psi) = X$.

Complexity

Recall that the problem whether a given game graph $\mathcal{G}$ with a compact description for Win admits a consistent winning strategy for Player 0, is NP-complete.

Complexity

Recall that the problem whether a given game graph $\mathcal{G}$ with a compact description for Win admits a consistent winning strategy for Player 0, is NP-complete.

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

Theorem. Let L be any extension of first-order logic 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.

Complexity

Recall that the problem whether a given game graph $\mathcal{G}$ with a compact description for Win admits a consistent winning strategy for Player 0, is NP-complete.

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

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

In fact, the problem is NEXPTIME-complete, even in the case where $\mathfrak{A}$ is just the set $\{0, 1\}$ and $X = \{\varnothing\}$.

Example: 3-colourability

$X(G) = \{s : (\text{edge,node}) \mapsto (e, u) : e \text{ is an edge of } G \text{ and } u \in e\}$

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

Claim. G is 3-colourable $\iff$ $V \cup E \models_{X(G)} \psi(\text{edge,node})$

Example: 3-colourability

$X(G) = \{s : (\text{edge,node}) \mapsto (e,u) : e \text{ is an edge of } G \text{ and } u \in e\}$

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

Claim. G is 3-colourable $\iff$ $V \cup E \models_{X(G)} \psi(\text{edge,node})$

A consistent winning strategy $S = (W,F)$ with $\text{Team}(S,\psi) = X(G)$ selects, for every assignment $s : (\text{edge,node}) \mapsto (e,u)$ at least one colour $c(s)$.

First conjunct: at most three colours are used

Second conjunct: the colour of (e,u) only depends on the node u

Final conjunct: the edge and the colour determine the node, i.e. a different colour is assigned to (e,u) and (e,v) for every edge $e = \{u,v\}$.

Inclusion logic

Inclusion logic: FO + inclusion atoms $\overline{x} \subseteq \overline{y}$.

Recall that $\mathfrak{A} \models_X \overline{x} \subseteq \overline{y}$ if for all $s \in X$ there exists a $t \in X$ with $t(\overline{y}) = s(\overline{x})$.

Inclusion logic is an interesting variant of a dependence logic, with somewhat unusual properties.

Contrary to dependence logic, it is not closed under subteams, but it is closed under unions of teams: If $\mathfrak{A} \models_X \psi$ and $\mathfrak{A} \models_Y \psi$ then also $\mathfrak{A} \models_{X \cup Y} \psi$

Inclusion logic is closely related to (a fragment of) least fixed-point logic (LFP).

Example: Infinite descending chains

$$(A, <) \models \exists x \exists y (y < x \wedge y \subseteq x) \quad \Longleftrightarrow \quad (A, <) \text{ is not well-founded}$$

Example: Infinite descending chains

$$(A, <) \models \exists x \exists y (y < x \wedge y \subseteq x) \quad \Longleftrightarrow \quad (A, <) \text{ is not well-founded}$$

A consistent winning strategy $S = (W, F)$ for this sentence just amounts to a selection of a team X of assignments $s : (x, y) \mapsto (a, b)$.

Example: Infinite descending chains

$$(A, <) \models \exists x \exists y (y < x \wedge y \subseteq x) \quad \Longleftrightarrow \quad (A, <) \text{ is not well-founded}$$

A consistent winning strategy $S = (W, F)$ for this sentence just amounts to a selection of a team X of assignments $s : (x, y) \mapsto (a, b)$.

All atoms are reachable by the opponent. Hence $\mathrm{Team}(S, y < x) = \mathrm{Team}(S, y \subseteq x) = X$. Thus, consistency of S means:

Example: Infinite descending chains

$$(A, <) \vDash \exists x \exists y (y < x \wedge y \subseteq x) \quad \Longleftrightarrow \quad (A, <) \text{ is not well-founded}$$

A consistent winning strategy $S = (W, F)$ for this sentence just amounts to a selection of a team X of assignments $s : (x, y) \mapsto (a, b)$.

All atoms are reachable by the opponent. Hence $\mathrm{Team}(S, y < x) = \mathrm{Team}(S, y \subseteq x) = X$. Thus, consistency of S means:

$(A, <) \vDash_X (y < x)$: For all $s : (x, y) \mapsto (a, b)$ in X, we have that $b < a$

Example: Infinite descending chains

$$(A, <) \models \exists x \exists y (y < x \wedge y \subseteq x) \quad \Longleftrightarrow \quad (A, <) \text{ is not well-founded}$$

A consistent winning strategy $S = (W, F)$ for this sentence just amounts to a selection of a team X of assignments $s : (x, y) \mapsto (a, b)$.

All atoms are reachable by the opponent. Hence $\mathrm{Team}(S, y < x) = \mathrm{Team}(S, y \subseteq x) = X$. Thus, consistency of S means:

$(A, <) \models_X (y < x)$: For all $s : (x, y) \mapsto (a, b)$ in X, we have that $b < a$

$(A, <) \models_X (y \subseteq x)$: For all $s : (x, y) \mapsto (a, b)$ in X, we have another assignment $t : (x, y) \mapsto (b, c)$ in X.

Example: Infinite descending chains

$$(A, <) \models \exists x \exists y (y < x \wedge y \subseteq x) \quad \Longleftrightarrow \quad (A, <) \text{ is not well-founded}$$

A consistent winning strategy $S = (W, F)$ for this sentence just amounts to a selection of a team X of assignments $s : (x, y) \mapsto (a, b)$.

All atoms are reachable by the opponent. Hence $\mathrm{Team}(S, y < x) = \mathrm{Team}(S, y \subseteq x) = X$. Thus, consistency of S means:

$(A, <) \models_X (y < x)$: For all $s : (x, y) \mapsto (a, b)$ in X, we have that $b < a$

$(A, <) \models_X (y \subseteq x)$: For all $s : (x, y) \mapsto (a, b)$ in X, we have another assignment $t : (x, y) \mapsto (b, c)$ in X.

Hence we have a consistent winning strategy in the model-checking game if, and only if, $(A, <)$ has an infinite descending chain.

Independence versus Henkin quantifiers

Independence atoms can be used to describe semantics of Henkin quantifiers.

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

$(A, P) \models \varphi$ if there exist functions $f, g : A \to A$ such that $\mathfrak{A} \models P(a, fa, c, gc)$ for all $a, c \in A$.

Independence versus Henkin quantifiers

Independence atoms can be used to describe semantics of Henkin quantifiers.

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

$(A, P) \models \varphi$ if there exist functions $f, g : A \to A$ such that $\mathfrak{A} \models P(a, fa, c, gc)$ for all $a, c \in A$.

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

Game-theoretic semantics: $(A, P) \models \psi$ if there exist functions $F : A \to \mathcal{P}(A) \setminus \{\varnothing\}$ and $G : A \times A \times A \to \mathcal{P}(A) \setminus \{\varnothing\}$ such that the team

$$X_{FG} = \{s : (x, y, u, v) \mapsto (a, b, c, d) : \quad b \in F(a) \text{ and } d \in G(a, b, c)\}$$

satisfies $xy \perp_u v$ and $Pxyuv$

Claim. The two formulae are equivalent.

Independence versus Henkin quantifiers

$$\varphi := \begin{pmatrix} \forall x & \exists y \\ \forall u & \exists v \end{pmatrix} Pxyuv \quad \models \quad \psi := \forall x \exists y \forall u \exists v (xy \perp_u v \wedge Pxyuv)$$

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 team X_{FG} then consists of all assignments (a, fa, c, gc) which clearly satisfies $Pxyuv$ and also satisfies the independence atom $xy \perp_u v$ since v is constant for any fixed value for u.

Independence versus Henkin quantifiers

$$\psi := \forall x \exists y \forall u \exists v (xy \perp_u v \wedge Pxyuv) \quad \models \quad \varphi := \begin{pmatrix} \forall x & \exists y \\ \forall u & \exists v \end{pmatrix} Pxyuv$$

Given a consistent winning strategy for ψ on $\mathfrak{A}$, witnessed by F and G, define Skolem functions for φ via a choice function ε on $\mathcal{P}(A)$.
Set $fa := \varepsilon F(a)$ and $gc := \varepsilon(\bigcup_{a \in A} G(a, fa, c))$.

Claim. $P(a, fa, c, gc)$ for all $a, c \in A$.

Prove that, for all a, c, the assignment (a, fa, c, gc) belongs to X_{FG}.

(1) Given a, c, there exists $a' \in A$ such that $gc \in G(a', fa', c)$; hence $(a', fa', c, gc) \in X_{FG}$.

(2) $(a, fa, c, d) \in X_{FG}$ for all $d \in G(a, fa, c)$.

(3) By the independence atom we infer that X_{FG} also contains (a, fa, c, gc).

Deterministic versus nondeterministic strategies

Our notion of consistent winning strategies is nondeteministic:

A consistent winning strategy of Player 0 for $\mathcal{G}$ and Win from X is a pair $S = (W, F) \subseteq (V, E)$ with $F \subseteq (W \times W) \cap E$ such that

(1) W is the set of nodes that are reachable from X via edges in F

(2) if $v \in V_0 \cap W$ then $vF \neq \emptyset$

(3) if $v \in V_1 \cap W$ then $vF = vE$

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

Deterministic versus nondeterministic strategies

The deterministic variant:

A deterministic consistent winning strategy of Player 0 for $\mathcal{G}$ and Win from X is a pair $S = (W, F) \subseteq (V, E)$ with $F \subseteq (W \times W) \cap E$ such that

(1) W is the set of nodes that are reachable from X via edges in F

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

(3) if $v \in V_1 \cap W$ then $vF = vE$

(4) $W \cap T \in \mathrm{Win}$.

In most classical games, deterministic strategies are no less powerful than nondeterministic ones. Is this also the case for second-order reachability games?

Why is the nondeterministic semantics the right one ?

Consider the formula $\exists x(y \subseteq x \wedge z \subseteq x)$ which says

> *The team under consideration can be extended by values for x such that all values for y and z in the team occur als values for x*

Why is the nondeterministic semantics the right one ?

Consider the formula $\exists x(y \subseteq x \wedge z \subseteq x)$ which says

> *The team under consideration can be extended by values for x such that all values for y and z in the team occur als values for x*

True for any team X: assign to x all values occurring for y and z in X.

Why is the nondeterministic semantics the right one ?

Consider the formula $\exists x(y \subseteq x \wedge z \subseteq x)$ which says

> *The team under consideration can be extended by values for x such that all values for y and z in the team occur als values for x*

True for any team X: assign to x all values occurring for y and z in X.

But under deterministic (strict) semantics for $(\exists x)$ this not the case. Let $X = \{s\}$ with $s(y) \neq s(z)$. By chosing just one witness for x, we cannot make the formula true for this team.

So under deterministic semantics, this formula says something different.

Why is the nondeterministic semantics the right one ?

Consider the formula $\exists x(y \subseteq x \wedge z \subseteq x)$ which says

> *The team under consideration can be extended by values for x such that all values for y and z in the team occur als values for x*

True for any team X: assign to x all values occurring for y and z in X.

But under deterministic (strict) semantics for $(\exists x)$ this not the case. Let $X = \{s\}$ with $s(y) \neq s(z)$. By chosing just one witness for x, we cannot make the formula true for this team.

So under deterministic semantics, this formula says something different.

So what?

Why is the nondeterministic semantics the right one ?

Consider the formula $\exists x(y \subseteq x \wedge z \subseteq x)$ and the team

$X =$

y	z	u
1	2	3
1	2	4

Clearly $X \models \exists x(y \subseteq x \wedge z \subseteq x)$ even under deterministic semantics.

Why is the nondeterministic semantics the right one ?

Consider the formula $\exists x(y \subseteq x \wedge z \subseteq x)$ and the team

$X =$

y	z	u
1	2	3
1	2	4

Clearly $X \models \exists x(y \subseteq x \wedge z \subseteq x)$ even under deterministic semantics.

But under deterministic semantics $X \upharpoonright \{y, z\} \not\models \exists x(y \subseteq x \wedge z \subseteq x)$

Why is the nondeterministic semantics the right one ?

Consider the formula $\exists x(y \subseteq x \wedge z \subseteq x)$ and the team

$X =$

y	z	u
1	2	3
1	2	4

Clearly $X \models \exists x(y \subseteq x \wedge z \subseteq x)$ even under deterministic semantics.

But under deterministic semantics $X \upharpoonright \{y, z\} \not\models \exists x(y \subseteq x \wedge z \subseteq x)$

Hence deterministic semantics violates the locality principle !

Downwards Closure

In model-checking games for dependence logic, deterministic strategies suffice. The reason is that dependence logic is downwards closed for teams.

A winning condition $\mathrm{Win} \subseteq \mathcal{P}(T)$ is downwards closed if $U \in \mathrm{Win}$ and $Z \subseteq U$ imply $Z \in \mathrm{Win}$.

Proposition. Let Win be downwards closed. Then Player 0 has a consistent winning strategy for G and Win if, and only if, she has a deterministic one.

Downwards Closure

In model-checking games for dependence logic, deterministic strategies suffice. The reason is that dependence logic is downwards closed for teams.

A winning condition $\mathrm{Win} \subseteq \mathcal{P}(T)$ is downwards closed if $U \in \mathrm{Win}$ and $Z \subseteq U$ imply $Z \in \mathrm{Win}$.

Proposition. Let Win be downwards closed. Then Player 0 has a consistent winning strategy for G and Win if, and only if, she has a deterministic one.

Is this sufficient condition also necessary?

Downwards Closure

In model-checking games for dependence logic, deterministic strategies suffice. The reason is that dependence logic is downwards closed for teams.

A winning condition $\mathrm{Win} \subseteq \mathcal{P}(T)$ is downwards closed if $U \in \mathrm{Win}$ and $Z \subseteq U$ imply $Z \in \mathrm{Win}$.

Proposition. Let Win be downwards closed. Then Player 0 has a consistent winning strategy for G and Win if, and only if, she has a deterministic one.

Is this sufficient condition also necessary?

No, but we can find a weaker condition, that is both necessary and sufficient for guaranteeing the possibility to elminate nondeterministic strategies.

Splits

A collection $\mathcal{F} \subseteq \mathcal{P}(T)$ has a split if there exist $U_1, U_2 \notin \mathcal{F}$ such that $U_1 \cup U_2 \in \mathcal{F}$.

Splits

A collection $\mathcal{F} \subseteq \mathcal{P}(T)$ has a split if there exist $U_1, U_2 \notin \mathcal{F}$ such that $U_1 \cup U_2 \in \mathcal{F}$.

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

Splits

A collection $\mathcal{F} \subseteq \mathcal{P}(T)$ has a split if there exist $U_1, U_2 \notin \mathcal{F}$ such that $U_1 \cup U_2 \in \mathcal{F}$.

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

Theorem. If $\mathrm{Win} \subseteq \mathcal{P}(T)$ has no splits, then for any second-order reachability game (G, Win), Player 0 has a consistent winning strategy, if and only if, she has a deterministic one. Conversely, if $\mathrm{Win} \subseteq \mathcal{P}(T)$ has a split, then there exists a game graph G with T as its set of terminal nodes such that Player 0 has a consistent winning strategy for (G, Win) but not a deterministic one.

Inclusion logic and least fixed-point logic

For inclusion logic, the second-order reachability games describing the model-checking problem can be translated into equivalent safety games.

Inclusion logic and least fixed-point logic

For inclusion logic, the second-order reachability games describing the model-checking problem can be translated into equivalent safety games.

Safety games also are the model-checking games for fixed-point formula, that only make use of greatest fixed points: the posGFP-fragment of least fixed-point logic LFP. Further, fundamental game-theoretic notions for safety games, such as winning regions and traps can be defined in both posGFP and inclusion logic.

Inclusion logic and least fixed-point logic

For inclusion logic, the second-order reachability games describing the model-checking problem can be translated into equivalent safety games.

Safety games also are the model-checking games for fixed-point formula, that only make use of greatest fixed points: the posGFP-fragment of least fixed-point logic LFP. Further, fundamental game-theoretic notions for safety games, such as winning regions and traps can be defined in both posGFP and inclusion logic.

By means of an interpretation argument one can then provide translations between the two logics.

Theorem. (Galliani and Hella) Inclusion logic ≡ posGFP.

Corollary. On finite structures, inclusion logic ≡ LFP.
In particular, on ordered finite structures, inclusion logic captures PTIME.

Part III: The Quest for a Logic for PTIME: Fixed-Point Logic, Inclusion Logic, and Counting

- Is there a logic for polynomial time?
- Least fixed point logic
- Inclusion logic versus fixed-point logic
- Fixed-point logic with counting
- Counting and Team semantics
- Threshold games and traps
- Inclusion logic with counting versus fixed-point logic with counting

The most important problem of Finite Model Theory

Is there a logic that captures PTIME?

The most important problem of Finite Model Theory

Is there a logic that captures PTIME?

Informal definition: A logic L captures PTIME if it defines precisely those properties of finite structures that are decidable in polynomial time:

(1) For every sentence $\psi \in L$, the set of finite models of ψ is decidable in polynomial time.

(2) For every PTIME-property S of finite τ-structures, there is a sentence $\psi \in L$ such that $S = \{\mathfrak{A} \in \mathrm{Fin}(\tau) : \mathfrak{A} \models \psi\}$.

The most important problem of Finite Model Theory

Is there a logic that captures PTIME?

Informal definition: A logic L captures PTIME if it defines precisely those properties of finite structures that are decidable in polynomial time:

(1) For every sentence $\psi \in L$, the set of finite models of ψ is decidable in polynomial time.

(2) For every PTIME-property S of finite τ-structures, there is a sentence $\psi \in L$ such that $S = \{\mathfrak{A} \in \mathrm{Fin}(\tau) : \mathfrak{A} \models \psi\}$.

The precise definition is more subtle. It includes certain effectiveness requirements to exclude pathological 'solutions'.

First-Order Logic

First-order logic (FO) is far too weak to capture PTIME.

- FO can express only local properties of finite structures
  Theorems of Gaifman and Hanf

  Global properties (e.g. planarity of graphs) are not expressible.

- FO has no mechanism for recursion or unbounded iteration.

  Transitive closures, reachability or termination properties, winning regions in games, etc. are not FO-definable.

- FO can only express properties in AC^0

  AC^0 is constant parallel time with polynomial hardware. In particular, FO $\subseteq$ LOGSPACE.

Second-Order Logic

Second-order logic (SO) is (probably) too strong to capture PTIME.

Fagin's Theorem. Existential SO captures NP.

Corollary. SO captures the polynomial hierarchy.

Thus SO captures polynomial time if, and only if, P = NP.

Second-Order Logic

Second-order logic (SO) is (probably) too strong to capture PTIME.

Fagin's Theorem. Existential SO captures NP.

Corollary. SO captures the polynomial hierarchy.

Thus SO captures polynomial time if, and only if, P = NP.

Monadic second-order logic is orthogonal to PTIME:

On words, MSO captures the regular languages, and not all PTIME-languages are regular.

On graphs, MSO can express NP-complete properties, such as 3-colourability.

Least fixed point logic LFP

Syntax. LFP extends FO by fixed point rule:

- For every formula $\psi(T, x_1 \dots x_k) \in \mathrm{LFP}[\tau \cup \{T\}]$,
  T k-ary relation variable, occuring only positive in ψ,
  build formulae $[\mathbf{lfp}\, T\overline{x} \,.\, \psi](\overline{x})$ and $[\mathbf{gfp}\, T\overline{x} \,.\, \psi](\overline{x})$

Semantics. On τ-structure $\mathfrak{A}$, $\psi(T, \overline{x})$ defines monotone operator

$$\begin{aligned} \psi^{\mathfrak{A}} : \mathcal{P}(A^k) &\longrightarrow \mathcal{P}(A^k) \\ T &\longmapsto \{\overline{a} : (\mathfrak{A}, T) \vDash \psi(T, \overline{a})\} \end{aligned}$$

- $\mathfrak{A} \vDash [\mathbf{lfp}\, T\overline{x} \,.\, \psi(T, \overline{x})](\overline{a}) \;:\Longleftrightarrow\; \overline{a} \in \mathbf{lfp}(\psi^{\mathfrak{A}})$
  $\mathfrak{A} \vDash [\mathbf{gfp}\, T\overline{x} \,.\, \psi(T, \overline{x})](\overline{a}) \;:\Longleftrightarrow\; \overline{a} \in \mathbf{gfp}(\psi^{\mathfrak{A}})$

LFP-definability for reachability and safety games

Reachability: Player 0 has winning strategy for game $\mathcal{G}$ from position v

$$\Longleftrightarrow$$

$$\mathcal{G} = (V, V_0, V_1, E) \models [\mathbf{lfp}\, Wx\,.\,(V_0x \wedge \exists y(Exy \wedge Wy)) \vee (V_1x \wedge \forall y(Exy \to Wy)](v)$$

Safety: Player 0 can avoid losing $\mathcal{G}$ from position v

$$\Longleftrightarrow$$

$$\mathcal{G} = (V, V_0, V_1, E) \models [\mathbf{gfp}\, Wx\,.\,(V_0x \to \exists y(Exy \wedge Wy)) \wedge (V_1x \to \forall y(Exy \to Wy)](v)$$

LFP-definability for reachability and safety games

Reachability: Player 0 has winning strategy for game $\mathcal{G}$ from position v

$$\Longleftrightarrow$$

$$\mathcal{G} = (V, V_0, V_1, E) \models [\mathbf{lfp}\, Wx\,.\,(V_0x \wedge \exists y(Exy \wedge Wy)) \vee (V_1x \wedge \forall y(Exy \to Wy)](v)$$

Safety: Player 0 can avoid losing $\mathcal{G}$ from position v

$$\Longleftrightarrow$$

$$\mathcal{G} = (V, V_0, V_1, E) \models [\mathbf{gfp}\, Wx\,.\,(V_0x \to \exists y(Exy \wedge Wy)) \wedge (V_1x \to \forall y(Exy \to Wy)](v)$$

On finite structure, the problem of computing winning regions for reachability (or safety) games is complete for LFP (via quantifier-free reductions)

LFP and polynomial time

Theorem (Immerman, Vardi)
On ordered finite structures, LFP captures PTIME.

On arbitrary finite structures, LFP can express certain PTIME-complete problems (such as winning regions of reachability and safety games), but fails to express all of PTIME.

LFP and polynomial time

Theorem (Immerman, Vardi)
On ordered finite structures, LFP captures PTIME.

On arbitrary finite structures, LFP can express certain PTIME-complete problems (such as winning regions of reachability and safety games), but fails to express all of PTIME.

LFP is unable to count.

For instance the class of graphs with an even number of vertices is not LFP-definable.

Further, LFP has a 0-1-law.

The gfp-fragment of LFP

The fragment posGFP of least fixed-point logic can be defined in two equivalent ways:

(1) posGFP is the closure of the set of formulae of form $[\mathbf{gfp}\, R\overline{x}\,.\,\varphi(R,\overline{x})](\overline{x})$, where $\varphi(R\overline{x})$ is in FO, under disjunction, conjunction, quantifiers, and applications of gfp.

(2) posGFP is the set of relations definable by simultaneous greatest fixed points of systems of first-order formulae.

The gfp-fragment of LFP

The fragment posGFP of least fixed-point logic can be defined in two equivalent ways:

(1) posGFP is the closure of the set of formulae of form $[\mathbf{gfp}\, R\overline{x}\,.\,\varphi(R,\overline{x})](\overline{x})$, where $\varphi(R\overline{x})$ is in FO, under disjunction, conjunction, quantifiers, and applications of gfp.

(2) posGFP is the set of relations definable by simultaneous greatest fixed points of systems of first-order formulae.

It is well-known that these two formulations are equivalent.

The alternation hierarchy

Fixed-point formulae become difficult to read and evaluate if they involve more than (very) few alternations of least and greatest fixed points.

Alternation between lfp- and gfp-operations define a hierarchy analogous to the Σ/Π hierarchies in first-order and second-order logic.

The fragment posGFP is at the bottom level of the alternation hierarchy of least fixed-point logic.

The alternation hierarchy

Fixed-point formulae become difficult to read and evaluate if they involve more than (very) few alternations of least and greatest fixed points.

Alternation between lfp- and gfp-operations define a hierarchy analogous to the Σ/Π hierarchies in first-order and second-order logic.

The fragment posGFP is at the bottom level of the alternation hierarchy of least fixed-point logic.

Theorem. (Immerman)
On finite structures the alternation hierarchy collapses: LFP $\equiv$ posGFP.

However, this is not the case in general. For instance on $(\mathbb{N}, +, \cdot)$ the alternation hierarchy of LFP is strict.

Model-checking games for LFP and posGFP

The model-checking games for LFP are parity games. It is, in the general case, not known whether these can be solved in polynomial time.

The model-checking games for posGFP are much simpler. They are safety games in which all infinite plays are won by the Verifier.

Model-checking games for LFP and posGFP

The model-checking games for LFP are parity games. It is, in the general case, not known whether these can be solved in polynomial time.

The model-checking games for posGFP are much simpler. They are safety games in which all infinite plays are won by the Verifier.

In fact, also the model-checking games for inclusion logic can be translated into safety games.

Safety games for inclusion logic

Let $\mathcal{G}(\mathfrak{A}, \psi)$ be a second-order reachability game for a formula $\psi \in \mathrm{Inc}$.
We define an associated safety game $\mathcal{H}(\mathfrak{A}, \psi)$ as follows:

From positions $(\overline{x} \subseteq \overline{y}, s)$, Verifier can move to any position $(\overline{x} \subseteq \overline{y}, t)$ such that $t(\overline{y}) = s(\overline{x})$. From there, Falsifier can make an arbitrary move upwards in the game forest.

From a given set $X \subseteq I$ of initial positions, Verifier must make sure to avoid
- terminal positions (φ, s), where φ is a first-order literal with $\mathfrak{A} \models_s \varphi$, and
- initial positions outside X

Safety games for inclusion logic

Let $\mathcal{G}(\mathfrak{A}, \psi)$ be a second-order reachability game for a formula $\psi \in \mathrm{Inc}$.
We define an associated safety game $\mathcal{H}(\mathfrak{A}, \psi)$ as follows:

From positions $(\overline{x} \subseteq \overline{y}, s)$, Verifier can move to any position $(\overline{x} \subseteq \overline{y}, t)$ such that $t(\overline{y}) = s(\overline{x})$. From there, Falsifier can make an arbitrary move upwards in the game forest.

From a given set $X \subseteq I$ of initial positions, Verifier must make sure to avoid
- terminal positions (φ, s), where φ is a first-order literal with $\mathfrak{A} \models_s \varphi$, and
- initial positions outside X

Proposition. Every consistent winning strategy $S = (W, F)$ from X for $\mathcal{G}(\mathfrak{A}, \psi)$ is also a winning strategy for $\mathcal{H}(\mathfrak{A}, \psi)$ that avoids $I \setminus X$, and vice versa.

Inclusion logic and least fixed-point logic

Theorem. (Galliani and Hella) For every formula $\varphi(\overline{z}) \in$ Inc one can construct a formula $\psi(X, \overline{z})$ in posGFP, and vice versa, such that, for all $\mathfrak{A}$ and all X

$$\mathfrak{A} \models_X \varphi \quad \Longleftrightarrow \quad (\mathfrak{A}, X) \models \forall \overline{z}(X\overline{z} \to \psi(X, \overline{z}))$$

Thus the maximal team satisfying φ coincides with $\mathbf{gfp}(\psi)$.

For the case of sentences, ψ and φ are equivalent.

Corollary. For sentences, inclusion logic and posGFP have the same expressive power.

Corollary. On finite structures, inclusion logic and LFP have the same expressive power. In particular, on ordered finite structures, inclusion logic captures PTIME.

Inclusion logic and least fixed-point logic

Theorem. (Galliani and Hella) For every formula $\varphi(\overline{z}) \in$ Inc one can construct a formula $\psi(X, \overline{z})$ in posGFP, and vice versa, such that, for all $\mathfrak{A}$ and all X

$$\mathfrak{A} \models_X \varphi \quad \Longleftrightarrow \quad (\mathfrak{A}, X) \models \forall \overline{z}(X\overline{z} \to \psi(X, \overline{z}))$$

Thus the maximal team satisfying φ coincides with $\mathbf{gfp}(\psi)$.

For the case of sentences, ψ and φ are equivalent.

Corollary. For sentences, inclusion logic and posGFP have the same expressive power.

Corollary. On finite structures, inclusion logic and LFP have the same expressive power. In particular, on ordered finite structures, inclusion logic captures PTIME.

However, without a linear order, both logics fail to capture polynomial time. In particular, neither LFP nor inclusion logic can express queries that involve counting.

Logic and counting: a difficult relationship

Counting is an important, computationally simple task that, however, is somewhat problematic for logic.

> *Ravens, so we read, can only count up to seven. They can't tell the difference between two numbers greater than or equal to eight. First-order logic is much the same as ravens, except that the cutoff point is rather higher: it's ω instead of 8.* *Wilfrid Hodges, Model Theory*

Logic and counting: a difficult relationship

Counting is an important, computationally simple task that, however, is somewhat problematic for logic.

> *Ravens, so we read, can only count up to seven. They can't tell the difference between two numbers greater than or equal to eight. First-order logic is much the same as ravens, except that the cutoff point is rather higher: it's ω instead of 8.* *Wilfrid Hodges, Model Theory*

In fact, for any fixed first-order sentence, the cutoff point is smaller than ω. Difficulties with counting are a central issue in finite model theory and concern also logics that are much stronger than first-order logic.

EVEN := $\{\mathfrak{A} : \mathfrak{A}$ has an even number of elements$\}$ is not definable in first-order logic FO, least fixed-point logic LFP, or the infinitary logic $L^{\omega}_{\infty\omega}$

Counting quantifiers

Add counting in one form or another to logical systems:

Simple variant of counting quantifiers, for fixed $i \in \mathbb{N}$:
$\exists^{\geq i} x$: there exist at least i elements x such that …

Such quantifiers do not increase the expressive power of first-order logic, but are nevertheless important for the study of bounded-variable logics.

Example. C^2 : two-variable first-order logic, with counting quantifiers $\exists^{\geq i}$.
- $C^2 \leq \mathrm{FO}$, but $C^2 \not\leq \mathrm{FO}^k$, for any fixed $k \in \omega$
- $\mathrm{Sat}(C^2)$ is decidable (Grädel, Otto, Rosen)

Counting quantifiers

Add counting in one form or another to logical systems:

Simple variant of counting quantifiers, for fixed $i \in \mathbb{N}$:
$\exists^{\geq i} x$: there exist at least i elements x such that ...

Such quantifiers do not increase the expressive power of first-order logic, but are nevertheless important for the study of bounded-variable logics.

Example. C^2 : two-variable first-order logic, with counting quantifiers $\exists^{\geq i}$.
- $C^2 \leq \mathrm{FO}$, but $C^2 \not\leq \mathrm{FO}^k$, for any fixed $k \in \omega$
- $\mathrm{Sat}(C^2)$ is decidable (Grädel, Otto, Rosen)

In many cases, and in particular in the quest for a logic for polynomial time, a more general form of counting is needed, where counting parameters are variables over a numeric domain.

Counting quantifiers over two-sorted structures

Expand a given (finite) structure $\mathfrak{A}$ by a second numeric sort

$$\mathfrak{A}^* = \mathfrak{A} \cup (\omega, <, +, \cdot, 0, 1) \quad \text{or} \quad \mathfrak{A}^* = \mathfrak{A} \cup (\{0, \ldots, |A|\}, <, 0, |A|)$$

Consider two-sorted logics with point variables $x, y, z, \ldots$ over A
and number variables $\mu, \nu, \lambda, \ldots$ over the numeric sort

Add counting quantifiers $\exists^{\geq \mu} x$ or counting terms $\#x[\varphi]$
Other counting quantifiers such as $\exists^{=\mu} x$ or $\exists^{\leq \mu} x$ etc. can also be used

Counting quantifiers over two-sorted structures

Expand a given (finite) structure $\mathfrak{A}$ by a second numeric sort

$$\mathfrak{A}^* = \mathfrak{A} \cup (\omega, <, +, \cdot, 0, 1) \quad \text{or} \quad \mathfrak{A}^* = \mathfrak{A} \cup (\{0, \ldots, |A|\}, <, 0, |A|)$$

Consider two-sorted logics with point variables $x, y, z, \ldots$ over A
and number variables $\mu, \nu, \lambda, \ldots$ over the numeric sort

Add counting quantifiers $\exists^{\geq \mu} x$ or counting terms $\#x[\varphi]$
Other counting quantifiers such as $\exists^{=\mu} x$ or $\exists^{\leq \mu} x$ etc. can also be used

Examples.

EVEN is defined by $\exists \mu \exists \nu (\nu + \nu = \mu \wedge \exists^{=\mu} x (x = x))$

A graph $G = (V, E)$ is regular if, and only if, $G \models \forall x \forall y (\#z[Exz] = \#z[Eyz])$

Fixed-point logic with counting

(FP + C): Two-sorted fixed-point logic with counting

If the numeric sort is $(\omega, <, +, \cdot)$, then numeric variables must be explicitly restricted to take only polynomially bounded values.

Least, greatest, or inflationary fixed point operators defining formulae of the form

$$[\mathbf{fp}\, R\overline{x\mu}_{\leq t} \,.\, \psi(R, \overline{x}, \overline{\mu})](\overline{y}, \overline{v}).$$

Fixed-point logic with counting

(FP + C): Two-sorted fixed-point logic with counting

If the numeric sort is $(\omega, <, +, \cdot)$, then numeric variables must be explicitly restricted to take only polynomially bounded values.

Least, greatest, or inflationary fixed point operators defining formulae of the form

$$[\mathbf{fp}\, R\overline{x\mu}_{\leq t} \,.\, \psi(R, \overline{x}, \overline{\mu})](\overline{y}, \overline{\nu}).$$

In the finite, it does not matter which kind of fixed-points. For us, it is convenient to consider only greatest fixed points, since these can be directly translated into existential second-order logic. To ensure monotonicity, use only counting quantifiers $\exists^{\geq \mu}$.

Fixed-point logic with counting versus polynomial time

Theorem. (FP+C) $\subsetneq$ PTIME (Cai, Fürer, Immerman 1992)

Fixed-point logic with counting versus polynomial time

Theorem. (FP+C) $\subsetneq$ PTIME (Cai, Fürer, Immerman 1992)

Although (FP+C) fails to capture PTIME it is the logic of reference in this area.

See survey by Dawar, SIGLOG-News, 2015

Fixed-point logic with counting versus polynomial time

Theorem. (FP+C) $\subsetneq$ PTIME (Cai, Fürer, Immerman 1992)

Although (FP+C) fails to capture PTIME it is the logic of reference in this area.
See survey by Dawar, SIGLOG-News, 2015

Fixed-point logic with counting captures PTIME on many interesting classes of structures, such as

- trees and forests (Immerman, Lander)
- planar graphs and graphs of bounded genus (Grohe)
- structures of bounded tree-width (Grohe, Marino)
- all classes of graphs that exclude a minor (Grohe)

Fixed-point logic with counting versus polynomial time

Theorem. (FP+C) $\subsetneq$ PTIME (Cai, Fürer, Immerman 1992)

Although (FP+C) fails to capture PTIME it is the logic of reference in this area.
See survey by Dawar, SIGLOG-News, 2015

Fixed-point logic with counting captures PTIME on many interesting classes of structures, such as

- trees and forests (Immerman, Lander)
- planar graphs and graphs of bounded genus (Grohe)
- structures of bounded tree-width (Grohe, Marino)
- all classes of graphs that exclude a minor (Grohe)

Current research investigates extensions of (FP+C), such as rank logic or Choiceless Polynomial Time as candidates for a logic that captures PTIME

Is counting relevant for logics with team semantics?

On the level of existential second-order logic (Σ^1_1), counting is available anyway!

- Σ^1_1 permits the introduction of a linear ordering $<$ on the universe.

- A second sort is not needed. Represent μ by the μ-th element of $<$

- $|X| \geq \mu$ if there is an injective function $F : \{a_1, \ldots, a_\mu\} \to A$

Is counting relevant for logics with team semantics?

On the level of existential second-order logic (Σ^1_1), counting is available anyway!

- Σ^1_1 permits the introduction of a linear ordering $<$ on the universe.

- A second sort is not needed. Represent μ by the μ-th element of $<$

- $|X| \geq \mu$ if there is an injective function $F : \{a_1, \ldots, a_\mu\} \to A$

Dependence atoms $=(\overline{x}, y)$ may also be seen as a form of counting.

Is counting relevant for logics with team semantics?

On the level of existential second-order logic (Σ^1_1), counting is available anyway!

- Σ^1_1 permits the introduction of a linear ordering $<$ on the universe.

- A second sort is not needed. Represent μ by the μ-th element of $<$

- $|X| \geq \mu$ if there is an injective function $F : \{a_1, \ldots, a_\mu\} \to A$

Dependence atoms $=(\overline{x}, y)$ may also be seen as a form of counting.

Hence counting is interesting in particular for logics that are weaker than Σ^1_1.
One motivation comes from the relationship between inclusion logic and LFP.

Counting in team semantics

There are several ways to add counting constructs to team semantics.

Here, we consider two-sorted structures $\mathfrak{A}^* = \mathfrak{A} \cup (\omega, <, +, \cdot)$ and teams of two-sorted assignments $s : (\overline{x}, \overline{\mu}) \mapsto (\overline{a}, \overline{n})$

Counting quantifiers:

$\mathfrak{A}^* \models_X \exists^{\geq \mu} x \varphi \quad :\Longleftrightarrow \quad \mathfrak{A}^* \models_{X[x \mapsto F]} \varphi$ for some function $F : X \to \mathcal{P}(A)$ with $|F(s)| \geq s(\mu)$ for all $s \in X$.

Counting quantifiers $\exists^{\geq \mu}$ preserve locality, closure under union and downwards closure. Instead quantifiers $\exists^{=\mu}$ or $\exists^{\leq \mu}$ are unsafe and may violate locality.

Proposition. Counting quantifiers $\exists^{\geq \mu}$ are expressible by means of dependence atoms: $\exists^{\geq \mu} y \varphi(\overline{x}, y) \equiv (\forall \lambda < \mu) \exists y (=(\overline{x} y, \lambda) \wedge \varphi(\overline{x}, y))$

Inclusion logic with counting

(Inc+C): First-order logic with team semantics, over two-sorted structures $\mathfrak{A}^* = \mathfrak{A} \cup (\omega, +, \cdot)$, possibly equipped with a set of functions $f : A^k \to \omega$, with inclusion atoms $(\overline{x\mu} \subseteq \overline{yv})$ and counting quantifiers $\exists^{\geq \mu}$.

Elementary properties: (Inc+C) is closed under union of teams

Question. Does the 'equivalence' between LFP and Inclusion Logic (on finite structures) extend to a similar 'equivalence' between (FP+C) and (Inc + C)?

Inclusion logic with Counting versus (FP+C)

The Galliani-Hella Theorem indeed generalizes to the result that for sentences, (FP+C) and inclusion logic with counting are equivalent.

Inclusion logic with Counting versus (FP+C)

The Galliani-Hella Theorem indeed generalizes to the result that for sentences, (FP+C) and inclusion logic with counting are equivalent.

Again, for open formulae, a more elaborate statement is needed.

Theorem. There are effective translations between formulae $\varphi(\overline{x\mu})$ of (Inc+C) and formulae $\psi(X, \overline{x\mu})$ in (FP+C), such that

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

Thus the maximal team satisfying φ coincides with $\mathbf{gfp}(\psi)$.

Inclusion logic with Counting versus (FP+C)

The Galliani-Hella Theorem indeed generalizes to the result that for sentences, (FP+C) and inclusion logic with counting are equivalent.

Again, for open formulae, a more elaborate statement is needed.

Theorem. There are effective translations between formulae $\varphi(\overline{x\mu})$ of (Inc+C) and formulae $\psi(X, \overline{x\mu})$ in (FP+C), such that

$$\mathfrak{A}^* \vDash_X \varphi(\overline{x\mu}) \quad \Longleftrightarrow \quad (\mathfrak{A}^*, X) \vDash \forall \overline{x\mu}(X\overline{x\mu} \to \psi(X, \overline{x\mu}))$$

Thus the maximal team satisfying φ coincides with $\mathbf{gfp}(\psi)$.

We prove this with a game-based approach.

As a first step, we define games that are appropriate for logics with counting.

Threshold games

Threshold game: $G = (V, E, t : V \to \omega)$

We can assume that $t(v) \leq \delta(v) + 1$ where $\delta(v) := |vE|$

At node v in a play, Player 0 selects $X \subseteq vE$ with $|X| \geq t(v)$

Player 1 choses $w \in X$ and the play proceeds from w.

When a player cannot move, she loses. This means that Player 0 loses at all nodes v with $\delta(v) < t(v)$ and Player 1 loses at nodes v with $t(v) = 0$.

Threshold safety games: Infinite plays are won by Player 0

Definability in threshold games

Winning regions of Player 0 in threshold safety games $G = (V, E, t : V \to \omega)$ are definable in fixed-point logic with counting

$$\text{Player 0 wins } G \text{ from } v \iff G \vDash [\mathbf{gfp}\, Wx \,.\, \exists^{\geq t(x)} y(Exy \wedge Wy)](v)$$

Further, the winning region for Player 0 is the maximal team W such that

$$G \vDash_W \exists^{\geq t(x)} y(Exy \wedge y \subseteq x)$$

Definability of I-traps

Let $\mathcal{G} = (V, E, t)$ be a threshold safety game with a set $I \subseteq V$ of initial positions. An I-trap in G is a set $Z \subseteq I$ such that Player 0 has a winning strategy that, moreover, avoids $I \setminus Z$.

Definability of I-traps

Let $\mathcal{G} = (V, E, t)$ be a threshold safety game with a set $I \subseteq V$ of initial positions. An I-trap in G is a set $Z \subseteq I$ such that Player 0 has a winning strategy that, moreover, avoids $I \setminus Z$.

Definability of I-traps in both logics:

$$Z \subseteq I \text{ is an I-trap in } \mathcal{G}$$

$$\Longleftrightarrow$$

$$(\mathcal{G}, I, Z) \models \forall x (Zx \to [\mathbf{gfp}\, Wx\,.\,(Ix \to Zx) \wedge \exists^{\geq t(x)} y (Exy \wedge Wy)](x))$$

$$\Longleftrightarrow$$

$$(\mathcal{G}, I) \models_Z \; Iz \wedge \exists x (z \subseteq x \wedge (\neg Ix \vee x \subseteq z) \wedge \exists^{\geq t(x)} y (Exy \wedge y \subseteq x))$$

Threshold games as model-checking games

For every formula $\psi(\overline{x}, \overline{\mu})$ in (Inc+C) and every structure $\mathfrak{A}^*$, one can construct a threshold safety game $\mathcal{T}(\mathfrak{A}^*, \psi)$ with the following properties:

- positions are (φ, s) where φ is a subformula of ψ and s is an assignment with domain free(φ).
- the set I of initial positions contains all pairs (ψ, s).
  Every team X for ψ defines the subset $I(X) := \{(\psi, s) : s \in X\}$ of I
- $\mathfrak{A}^* \models_X \psi(\overline{x}, \overline{\mu}) \iff I(X)$ is an I-trap in $\mathcal{T}(\mathfrak{A}^*, \psi)$

Threshold games as model-checking games

For every formula $\psi(\overline{x}, \overline{\mu})$ in (Inc+C) and every structure $\mathfrak{A}^*$, one can construct a threshold safety game $\mathcal{T}(\mathfrak{A}^*, \psi)$ with the following properties:

- positions are (φ, s) where φ is a subformula of ψ and s is an assignment with domain free(φ).
- the set I of initial positions contains all pairs (ψ, s).
  Every team X for ψ defines the subset $I(X) := \{(\psi, s) : s \in X\}$ of I
- $\mathfrak{A}^* \models_X \psi(\overline{x}, \overline{\mu}) \iff I(X)$ is an I-trap in $\mathcal{T}(\mathfrak{A}^*, \psi)$

Moreover, for every formula $\psi(\overline{x}, \overline{\mu})$ there is a first-order interpretation I_ψ that interprets the game $\mathcal{T}(\mathfrak{A}^*, \psi)$ in $\mathfrak{A}^*$.

Threshold games as model-checking games

For every formula $\psi(\overline{x}, \overline{\mu})$ in (Inc+C) and every structure $\mathfrak{A}^*$, one can construct a threshold safety game $\mathcal{T}(\mathfrak{A}^*, \psi)$ with the following properties:

- positions are (φ, s) where φ is a subformula of ψ and s is an assignment with domain free(φ).
- the set I of initial positions contains all pairs (ψ, s).
  Every team X for ψ defines the subset $I(X) := \{(\psi, s) : s \in X\}$ of I
- $\mathfrak{A}^* \models_X \psi(\overline{x}, \overline{\mu}) \iff I(X)$ is an I-trap in $\mathcal{T}(\mathfrak{A}^*, \psi)$

Moreover, for every formula $\psi(\overline{x}, \overline{\mu})$ there is a first-order interpretation I_ψ that interprets the game $\mathcal{T}(\mathfrak{A}^*, \psi)$ in $\mathfrak{A}^*$.

Analogous statements hold for fixed-point logic with counting.

Game-based translations between (Inc+C) and (FP+C)

Goal: Translate any $\psi(\overline{x\mu})$ in (Inc+C) to $\varphi(X, \overline{x\mu})$ in (FP+C) such that

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

Game-based translations between (Inc+C) and (FP+C)

Goal: Translate any $\psi(\overline{x\mu})$ in (Inc+C) to $\varphi(X, \overline{x\mu})$ in (FP+C) such that

$$\mathfrak{A}^* \models_X \psi(\overline{x\mu}) \iff (\mathfrak{A}^*, X) \models \forall\overline{x}\forall\overline{\mu}(X\overline{x\mu} \to \varphi(X, \overline{x\mu}))$$

Take a formula in (FP+C) that defines I-traps in threshold games:

$\mathcal{T}(\mathfrak{A}^*, \psi), Z \models \forall x(Zx \to \mathrm{itrap}(Z, x)) \iff Z$ is an I-trap in $\mathcal{T}(\mathfrak{A}^*, \psi)$.

Game-based translations between (Inc+C) and (FP+C)

Goal: Translate any $\psi(\overline{x\mu})$ in (Inc+C) to $\varphi(X, \overline{x\mu})$ in (FP+C) such that

$$\mathfrak{A}^* \models_X \psi(\overline{x\mu}) \iff (\mathfrak{A}^*, X) \models \forall\overline{x}\forall\overline{\mu}(X\overline{x\mu} \to \varphi(X, \overline{x\mu}))$$

Take a formula in (FP+C) that defines I-traps in threshold games:

$\mathcal{T}(\mathfrak{A}^*, \psi), Z \models \forall x(Zx \to \mathrm{itrap}(Z, x)) \iff Z$ is an I-trap in $\mathcal{T}(\mathfrak{A}^*, \psi)$.

The interpretation I_ψ defines a copy of the game $\mathcal{T}(\mathfrak{A}^*, \psi)$ inside $\mathfrak{A}^*$.
Further it maps formulae on the game back to formulae on $\mathfrak{A}^*$.

$$I_\psi : \mathrm{itrap}(Z, x) \mapsto \mathrm{itrap}^*(Y, \overline{yv}).$$

If Y is the set of tuples in $\mathfrak{A}^*$ associated with $Z \subseteq I$, then

$(\mathfrak{A}^*, Y) \models \forall\overline{yv}(Y\overline{yv} \to \mathrm{itrap}^*(Y, \overline{yv})) \iff Z$ is an I-trap in $\mathcal{T}(\mathfrak{A}^*, \psi)$.

Game-based translations between (Inc+C) and (FP+C)

If Y is the set of tuples in $\mathfrak{A}^*$ associated with $Z \subseteq I$, then

$(\mathfrak{A}^*, Y) \models \forall \overline{yv}(Y\overline{yv} \to \text{itrap}^*(Y, \overline{yv})) \iff Z$ is an I-trap in $\mathcal{T}(\mathfrak{A}^*, \psi)$.

Game-based translations between (Inc+C) and (FP+C)

If Y is the set of tuples in $\mathfrak{A}^*$ associated with $Z \subseteq I$, then

$(\mathfrak{A}^*, Y) \models \forall \overline{yv}(Y\overline{yv} \to \text{itrap}^*(Y, \overline{yv})) \iff Z$ is an I-trap in $\mathcal{T}(\mathfrak{A}^*, \psi)$.

The tuples describing positions (ψ, s) with $s \in X$ are definable in $(\mathfrak{A}^*, X)$.

We can thus massage $\text{itrap}^*(Y, \overline{yv})$ into $\varphi(X, \overline{x\mu})$ in (FP+C) such that

$$(\mathfrak{A}^*, X) \models \forall \overline{x\mu}(X\overline{x\mu} \to \varphi(X, \overline{x\mu})) \iff I(X) \text{ is an I-trap in } \mathcal{T}(\mathfrak{A}^*, \psi)$$
$$\iff \mathfrak{A} \models_X \psi(\overline{x\mu}).$$

Game-based translations between (Inc+C) and (FP+C)

If Y is the set of tuples in $\mathfrak{A}^*$ associated with $Z \subseteq I$, then
$(\mathfrak{A}^*, Y) \models \forall \overline{yv}(Y\overline{yv} \to \text{itrap}^*(Y, \overline{yv})) \iff Z$ is an I-trap in $\mathcal{T}(\mathfrak{A}^*, \psi)$.

The tuples describing positions (ψ, s) with $s \in X$ are definable in $(\mathfrak{A}^*, X)$.

We can thus massage $\text{itrap}^*(Y, \overline{yv})$ into $\varphi(X, \overline{x\mu})$ in (FP+C) such that

$$(\mathfrak{A}^*, X) \models \forall \overline{x\mu}(X\overline{x\mu} \to \varphi(X, \overline{x\mu})) \iff I(X) \text{ is an I-trap in } \mathcal{T}(\mathfrak{A}^*, \psi)$$
$$\iff \mathfrak{A} \models_X \psi(\overline{x\mu}).$$

An analogous argument works for the translation of posGFP into Inc.

Summary

There is a rich collection of logics of dependence and independence, on the basis of different atomic dependency properties

All these logics are based on team semantics

In terms of expressive power, logics of dependence and independence are equivalent to existential second-order logic or natural fragments of it.

Inclusion logic is equivalent to the posGFP fragment of fixed-point logic and captures polynomial time on ordered finite structures

The model-checking games for logics with team semantics are second-order reachability games. These game provide evaluation procedures and complexity results for these logics.

Summary

In the case of inclusion logic, the games can be simplified to classical safety games

The equivalence between positive greatest fixed point logic and inclusion logic lifts to an equivalence between corresponding counting extensions. Thus inclusion logic with counting is rather close to PTIME.

Future work: Counting may become very relevant but also very difficult once we shall move to multi-team semantics.