Dependency: the question-based view

Ivano Ciardelli

LoDE Workshop — 11 Aug 2020

What is dependency?

~~What is dependency?~~

~~What is dependency?~~

What are the fruitful ways to conceptualize the notion of dependency, and theorize about its logic?

~~What is dependency?~~

What are the fruitful ways to conceptualize the notion of dependency, and theorize about its logic?

In the dependence logic tradition, dependency is construed as a relation between variables (V-dependency).

~~What is dependency?~~

What are the fruitful ways to conceptualize the notion of dependency, and theorize about its logic?

In the dependence logic tradition, dependency is construed as a relation between variables (V-dependency).

I want to discuss a different conception, grounded in work on inquisitive logic: dependency as a relation between questions (Q-dependency).

V-dependency

Let T be a team—a set of assignments.

$$T = \begin{array}{|c|cccccc|} \hline \mathrm{x} & 1 & 1 & 2 & 2 & 3 & 3 \\ \mathrm{y} & 0 & 0 & 1 & 1 & 1 & 1 \\ \hline \end{array}$$

Meta-language

$$\mathbb{D}_T(x, y) \iff \forall g, g' \in T : g(x) = g'(x) \Rightarrow g(y) = g'(y)$$

Object language formula

$$T \models \mathsf{D}(x, y) \iff \mathbb{D}_T(x, y)$$

x	1	1	2	2	3	3
y	0	0	1	1	1	1

Equivalent formulation

$\mathbb{D}_T(x, y) \iff \forall T' \subseteq T : x$ constant in $T' \Rightarrow y$ constant in T'

x	1	1	2	2	3	3
y	0	0	1	1	1	1

Equivalent formulation

$$\begin{aligned}\mathbb{D}_T(x,y) &\iff \forall T' \subseteq T : x \text{ constant in } T' \Rightarrow y \text{ constant in } T' \\ &\iff \forall T' \subseteq T : T' \text{ settles what is the value of } x \Rightarrow \\ &\qquad\qquad\qquad\qquad T' \text{ settles what is the value of } y\end{aligned}$$

x	1	1	2	2	3	3
y	0	0	1	1	1	1

Equivalent formulation

$$\mathbb{D}_T(x,y) \iff \forall T' \subseteq T : x \text{ constant in } T' \Rightarrow y \text{ constant in } T'$$
$$\iff \forall T' \subseteq T : T' \text{ settles what is the value of } x \Rightarrow T' \text{ settles what is the value of } y$$

Support for questions

$T \models Q \iff Q$ is settled in T

Example: value questions

- Vx : what is the value of x?
- $T \models Vx \iff \forall g, g' \in T : g(x) = g'(x)$

x	1	1	1	1
y	2	2	3	3

$T \models Vx \qquad T \not\models Vy$

Q-dependency

Meta-language relation

$$Q \rightsquigarrow_T Q' \iff \forall T' \subseteq T : T' \models Q \Rightarrow T' \models Q'$$

Object language formula

$$T \models Q \to Q' \iff Q \rightsquigarrow_T Q'$$

Q-dependency

Meta-language relation

$$Q \rightsquigarrow_T Q' \iff \forall T' \subseteq T : T' \models Q \Rightarrow T' \models Q'$$

Object language formula

$$T \models Q \to Q' \iff Q \rightsquigarrow_T Q'$$

Dependency of value questions

$$\begin{aligned} Vx \rightsquigarrow_T Vy &\iff \forall T' \subseteq T : T' \models Vx \Rightarrow T' \models Vy \\ &\iff \forall T' \subseteq T : x \text{ constant in } T' \Rightarrow y \text{ constant in } T' \\ &\iff \mathbb{D}_T(x, y) \end{aligned}$$

We capture the same property of a team, but as a relation between questions, rather than as a relation between variables.

What are the reasons to be interested in this alternative perspective?

I'll discuss three:

1. flexibility
2. generality
3. logicality

1. Flexibility

Questions are sentences; as such, they can be combined by logical operators.

Two examples where this is convenient:

- Conjunction
- Conditionalization

Conjunction

Both dependency relations can be generalized naturally to the case of multiple determinants.

$$\mathbb{D}_T(x_1, \ldots, x_n; y) \iff \forall g, g' \in T : \text{if } g(x_i) = g'(x_i) \text{ for all } i \leq n \text{ then } g(y) = g'(y)$$

$$Q_1, \ldots, Q_n \rightsquigarrow_T Q' \iff \forall T' \subseteq T : \text{ if } T' \models Q_i \text{ for all } i \leq n \text{ then } T' \models Q'$$

In the case of Q-dependency, however, the general version is reducible to the binary version relying on conjunction.

Question conjunction

$T \models Q \wedge Q' \iff T \models Q$ and $T \models Q'$

Reduction

$Q_1, \ldots, Q_n \rightsquigarrow_T Q' \iff Q_1 \wedge \cdots \wedge Q_n \rightsquigarrow_T Q'$

Object language

In the object language we can make do with a binary connective $\rightarrow$:

$T \models Q_1 \wedge \cdots \wedge Q_n \rightarrow Q' \iff Q_1, \ldots, Q_n \rightsquigarrow_T Q'$

Conditionalization

Below, y is the Fibonacci number of x. Then y determines x provided $y \geq 2$.
What we have is a conditional dependency.

x	1	2	3	4	5	6	...
y	1	1	2	3	5	8	...

Conditionalization

Below, y is the Fibonacci number of x. Then y determines x provided $y \geq 2$.
What we have is a conditional dependency.

x	1	2	3	4	5	6	...
y	1	1	2	3	5	8	...

To capture this we could generalize the relation $\mathbb{D}$ even further:

$$\mathbb{D}_T(S;\, x_1, \ldots, x_n;\, y) \iff \forall g, g' \in T \cap |S| \quad \text{where } |S| = \{g \mid M \models_g S\}$$
$$\text{if } g(x_i) = g'(x_i) \text{ for all } i \leq n$$
$$\text{then } g(y) = g'(y)$$

Conditionalization

Below, y is the Fibonacci number of x. Then y determines x provided $y \geq 2$.
What we have is a conditional dependency.

x	1	2	3	4	5	6	...
y	1	1	2	3	5	8	...

To capture this we could generalize the relation $\mathbb{D}$ even further:

$$\mathbb{D}_T(S;\, x_1, \ldots, x_n;\, y) \iff \forall g, g' \in T \cap |S| \quad \text{where } |S| = \{g \mid M \models_g S\}$$
$$\text{if } g(x_i) = g'(x_i) \text{ for all } i \leq n$$
$$\text{then } g(y) = g'(y)$$

We then need a corresponding generalized operator $D(S; x_1, \ldots, x_n; y)$.

From the Q-dependency perspective, things are simpler.

Support for statements

$T \models S \iff T \subseteq |S| \qquad$ where $|S| = \{g \mid M \models_g S\}$

Then we need no special "slot" for statements in the dependency relation $\rightsquigarrow$. Statements and questions can be treated on a par as determinants:

$$
\begin{aligned}
S, Q \rightsquigarrow_T Q' \quad &\iff \quad \forall T' \subseteq T : T' \models S \text{ and } T' \models Q \Rightarrow T' \models Q' \\
&\iff \quad \forall T' \subseteq T : T' \subseteq |S| \text{ and } T' \models Q \Rightarrow T' \models Q' \\
&\iff \quad \forall T' \subseteq T \cap |S| : T' \models Q \Rightarrow T' \models Q' \\
&\iff \quad Q \rightsquigarrow_{T \cap |S|} Q'
\end{aligned}
$$

From the Q-dependency perspective, things are simpler.

Support for statements

$T \models S \iff T \subseteq |S| \qquad$ where $|S| = \{g \mid M \models_g S\}$

Then we need no special "slot" for statements in the dependency relation $\rightsquigarrow$. Statements and questions can be treated on a par as determinants:

$$\begin{aligned} S, Q \rightsquigarrow_T Q' &\iff \forall T' \subseteq T : T' \models S \text{ and } T' \models Q \Rightarrow T' \models Q' \\ &\iff \forall T' \subseteq T : T' \subseteq |S| \text{ and } T' \models Q \Rightarrow T' \models Q' \\ &\iff \forall T' \subseteq T \cap |S| : T' \models Q \Rightarrow T' \models Q' \\ &\iff Q \rightsquigarrow_{T \cap |S|} Q' \end{aligned}$$

We get conditional dependencies for free without generalizing the relation $\rightsquigarrow_T$, simply by plugging in statements as determinants.

Example

Our observation above amounts to the relation:

$$y \geq 2,\ Vy \rightsquigarrow_T Vx$$

x	1	2	3	4	5	6	...
y	1	1	2	3	5	8	...

Example

Our observation above amounts to the relation:

$$y \geq 2,\ Vy \rightsquigarrow_T Vx$$

x	1	2	3	4	5	6	...
y	1	1	2	3	5	8	...

In the object language, we can express this without additional resources as:

$$((y \geq 2) \wedge Vy) \rightarrow Vx$$

Example

Our observation above amounts to the relation:

$$y \geq 2,\ Vy \rightsquigarrow_T Vx$$

x	1	2	3	4	5	6	...
y	1	1	2	3	5	8	...

In the object language, we can express this without additional resources as:

$$((y \geq 2) \wedge Vy) \rightarrow Vx$$

or equivalently as

$$(y \geq 2) \rightarrow (Vy \rightarrow Vx)$$

Main point

The Q-dependency relation $\rightsquigarrow_T$ connects sentences.

Conjunction

- Sentences can be conjoined.
- We can use $\wedge$ to combine multiple determinants:

$$Q \wedge Q' \rightarrow Q''$$

Conditionalization

- Statements are also sentences.
- Having them as determinants gives us conditional dependencies for free.

$$S \wedge Q \rightarrow Q'$$

2. Generality

Example

x	3	3	4	4	5	5	6	6
y	2	4	3	5	4	6	5	7

No non-trivial V-dependencies hold:

- We have neither $\mathbb{D}_T(x, y)$ nor $\mathbb{D}_T(y, x)$.

2. Generality

Example

x	3	3	4	4	5	5	6	6
y	2	4	3	5	4	6	5	7

No non-trivial V-dependencies hold:

- We have neither $\mathbb{D}_T(x, y)$ nor $\mathbb{D}_T(y, x)$.

Yet, we can still recognize many interesting dependence patterns:

2. Generality

Example

x	3	3	4	4	5	5	6	6
y	2	4	3	5	4	6	5	7

No non-trivial V-dependencies hold:

- We have neither $\mathbb{D}_T(x, y)$ nor $\mathbb{D}_T(y, x)$.

Yet, we can still recognize many interesting dependence patterns:

- Whether x is prime determines whether it is even.

2. Generality

Example

x	3	3	4	4	5	5	6	6
y	2	4	3	5	4	6	5	7

No non-trivial V-dependencies hold:

- We have neither $\mathbb{D}_T(x, y)$ nor $\mathbb{D}_T(y, x)$.

Yet, we can still recognize many interesting dependence patterns:

- Whether x is prime determines whether it is even.
- Value of x together with whether $x < y$ determines the value of y.

2. Generality

Example

x	3	3	4	4	5	5	6	6
y	2	4	3	5	4	6	5	7

No non-trivial V-dependencies hold:

- We have neither $\mathbb{D}_T(x, y)$ nor $\mathbb{D}_T(y, x)$.

Yet, we can still recognize many interesting dependence patterns:

- Whether x is prime determines whether it is even.
- Value of x together with whether $x < y$ determines the value of y.
- ...

These can be captured as instances of the Q-dependence relation once we look beyond the special case of value questions.

These can be captured as instances of the Q-dependence relation once we look beyond the special case of value questions.

Polar questions

If S is a statement, we can form the polar question $?S$ ('whether S').

$$T \models ?S \iff S \text{ has the same truth-value at all } g \in T$$

Example

$$T \models ?\mathrm{Even}(x) \iff \forall g \in T : g(x) \text{ is even or } \forall g \in T : g(x) \text{ is odd}$$

The relations observed above can be captured as Q-dependencies:

x	3	3	4	4	5	5	6	6
y	2	4	3	5	4	6	5	7

Whether x is prime determines whether it is even

- Meta-language: $?\mathrm{Prime}(x) \rightsquigarrow_T ?\mathrm{Even}(x)$
- Object language: $?\mathrm{Prime}(x) \rightarrow ?\mathrm{Even}(x)$

Value of x together with whether $x < y$ determines value of y:

- Meta-language: $Vx,\ ?(x < y) \rightsquigarrow_T Vy$
- Object language: $Vx \wedge ?(x < y) \rightarrow Vy$

Another example

Value of x determines the odd prime factors of y.

x	1	1	2	2	3	3	4	4
y	2	4	3	6	5	10	15	30

Which questions

- $OF(z, y) =$ "z is an odd prime factor of y"
- $Wz.OF(z, y) =$ " what are the odd prime factors of y?"
- $T \models Wz.OF(z, y) \iff \forall g, g' \in T : OF[g(y)] = OF[g'(y)]$

Value of x determines the odd prime factors of y

- Meta-language: $Vx \rightsquigarrow_T Wz.OF(z, y)$
- Object language: $Vx \rightarrow Wz.OF(y, z)$

Main point

- We have a dependency whenever information of a type (input type) yields information of another type (output type).
- Questions can be seen as names for information types:
  - Vx : value of x
  - $?Even(x)$: parity of x
  - $WyOF(x, y)$: odd prime factors of x
- To represent an arbitrary dependence, we just have to find questions Q and Q' that capture the relevant input and output types.
- Then the dependency amounts to $Q \rightsquigarrow_T Q'$.

3. Logicality

Recall

$$A_1, \ldots, A_n \rightsquigarrow_T C \iff \forall T' \subseteq T : (T' \models A_i \text{ for all } i) \Rightarrow T' \models C$$

3. Logicality

Recall

$A_1, \ldots, A_n \rightsquigarrow_T C \iff \forall T' \subseteq T : (T' \models A_i \text{ for all } i) \Rightarrow T' \models C$

Observation

$\rightsquigarrow_T$ is a consequence relation (satisfies reflexivity, transitivity, and weakening). Moreover, for all sentences A, B, C we have:

$$A, B \rightsquigarrow_T C \iff A \wedge B \rightsquigarrow_T C \iff A \rightsquigarrow_T B \to C$$

Logical consequence

$\models \; := \bigcap_{M,T} (\rightsquigarrow_T)$

Logical consequence

$\models \; := \bigcap_{M,T}(\rightsquigarrow_T)$

Conservativity

In restriction to statements, $\models$ coincides with standard logical entailment:

$S \models S' \iff \forall M, g : (M \models_g S \Longrightarrow M \models_g S')$

Logical consequence

$\models \; := \bigcap_{M,T}(\rightsquigarrow_T)$

Conservativity

In restriction to statements, $\models$ coincides with standard logical entailment:
$S \models S' \iff \forall M, g : (M \models_g S \Longrightarrow M \models_g S')$

Logical dependency

$S, Q \models Q'$ expresses the fact that given S, Q logically determines Q'.

Examples

- $Px \leftrightarrow \neg Qy$, $?Px \models ?Qy$
- $y = f(x)$, $Vx \models Vy$

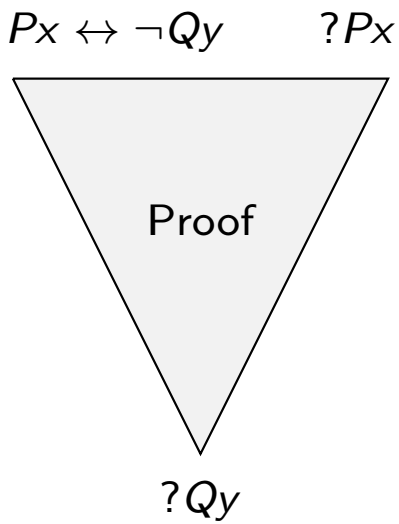
So, we can generalize the classical notion of entailment to questions: logical dependency turns out to be a facet of entailment.

So, we can generalize the classical notion of entailment to questions: logical dependency turns out to be a facet of entailment.

This also has many repercussions.

Example

Valid entailments are the sort of thing we prove in logic. If we have a proof system for this generalized entailment, we can prove logical dependencies.



Implication

The connective $\rightarrow$ behaves like an implication for this notion of entailment:
$A, B \models C \iff A \models B \rightarrow C$

Implication

The connective $\to$ behaves like an implication for this notion of entailment:
$A, B \models C \iff A \models B \to C$

This means that we can make inferences with dependence formulas by the standard rules for implication:

$$\frac{\begin{array}{c}[Q] \\ \vdots \\ Q'\end{array}}{Q \to Q'}\ (\to i) \qquad\qquad \frac{Q \quad Q \to Q'}{Q'}\ (\to e)$$

Implication

The connective $\to$ behaves like an implication for this notion of entailment:
$A, B \models C \iff A \models B \to C$

This means that we can make inferences with dependence formulas by the standard rules for implication:

$$\dfrac{\begin{array}{c}[Q] \\ \vdots \\ Q'\end{array}}{Q \to Q'}\ (\to i) \qquad\qquad \dfrac{Q \quad Q \to Q'}{Q'}\ (\to e)$$

So the connection between dependence formulas and implications is not just notational, but substantial.

Implication

The connective $\to$ behaves like an implication for this notion of entailment:
$A, B \models C \iff A \models B \to C$

This means that we can make inferences with dependence formulas by the standard rules for implication:

$$\dfrac{\begin{array}{c}[Q] \\ \vdots \\ Q'\end{array}}{Q \to Q'}\ (\to i) \qquad\qquad \dfrac{Q \quad Q \to Q'}{Q'}\ (\to e)$$

So the connection between dependence formulas and implications is not just notational, but substantial.

Conservativity

When S, S' are statements, $S \to S'$ coincides with the material conditional.

Main point

By viewing dependency as a relation between sentences, we can see that it is deeply connected to the central concerns of logic:

- logical dependency is a generalization to questions of logical entailment;
- the dependency operator $\rightarrow$ is a generalization of classical implication.

From approach to systems

From approach to systems

- My presentation has been vague: I have not defined a specific system.
- This was on purpose: what I wanted to present is a general approach to dependence, a template that can have many concrete realizations.
- The approach itself determines certain features of a system (team semantics, treatment of $\wedge, \rightarrow$) but leaves much to be specified.

From approach to systems

- My presentation has been vague: I have not defined a specific system.
- This was on purpose: what I wanted to present is a general approach to dependence, a template that can have many concrete realizations.
- The approach itself determines certain features of a system (team semantics, treatment of $\wedge, \rightarrow$) but leaves much to be specified.
- In fact, I should have been more vague: I have taken teams to be sets of assignments, but the approach works for any set of classical points of evaluation (propositional valuations, possible worlds, etc. ...).

Next, I want to look at a specific system: inquisitive first-order logic (InqBQ).

This brief incursion has three aims:

- illustrate the ideas above in a concrete setting;
- mention two results from Gianluca Grilletti's forthcoming dissertation;
- to advertise some major open questions.

Language

$\varphi ::= p \mid \bot \mid \varphi \wedge \varphi \mid \varphi \to \varphi \mid \forall x \varphi \mid \varphi \mathbin{\mathbb{\vee}} \varphi \mid \overline{\exists} x \varphi$ with p a FOL atom

Classical language

Formulas without $\mathbb{\vee}, \overline{\exists}$ are called classical and identified with FOL formulas, where $\neg, \vee, \exists$ are taken as defined operators:

- $\neg\varphi := \varphi \to \bot$
- $\varphi \vee \psi := \neg(\neg\varphi \wedge \neg\psi)$
- $\exists x \varphi := \neg\forall x \neg\varphi$

Inquisitive operators

We read the inquisitive operators $\mathbb{\vee}, \overline{\exists}$ as question-forming operators. E.g.:

- $?\alpha := \alpha \mathbin{\mathbb{\vee}} \neg\alpha \approx$ "whether α or $\neg\alpha$"
- $\overline{\exists} x \alpha(x) \approx$ "what is an x such that $\alpha(x)$"

Models

$M = \langle W, D, I \rangle$ where I equips each $w \in W$ with a rel. structure I_w over D.

In InqBQ we are interested in questions and dep. about what the world is like. Hence, the relevant 'team' is not a set of assignments, but a set of worlds.

Semantics

Relation $s \models_g \varphi$ where $s \subseteq W$ and $g : \mathrm{Var} \to D$.

Semantic clauses

- $s \models_g p \iff \forall w \in s : I_w \models_g p$
- $s \models_g \bot \iff s = \emptyset$
- $s \models_g \varphi \wedge \psi \iff s \models_g \varphi$ and $s \models_g \psi$
- $s \models_g \varphi \mathbin{\overline{\vee}} \psi \iff s \models_g \varphi$ or $s \models_g \psi$
- $s \models_g \varphi \to \psi \iff \forall t \subseteq s : t \models_g \varphi \Longrightarrow t \models_g \psi$
- $s \models_g \forall x \varphi \iff \forall d \in D : s \models_{g[x \mapsto d]} \varphi$
- $s \models_g \overline{\exists} x \varphi \iff \exists d \in D : s \models_{g[x \mapsto d]} \varphi$

Classical formulas are 'flat'

For classical formulas α, support amounts to global truth:
$s \models_g \alpha \iff \forall w \in s : I_w \models_g \alpha$ in standard FOL

Conservativity

Restricted to classical formulas, entailment coincides with FOL-entailment.

Questions

Using $\bigvee, \overline{\exists}$ we can express a broad repertoire of interesting questions.

Questions

Using $\mathbb{V}, \overline{\exists}$ we can express a broad repertoire of interesting questions.

Polar questions

- $?Pc \approx$ does c have property P?
- $s \models_g ?Pc \iff Pc$ has the same truth-value in all $w \in s$

Mention-all questions

- $\forall x?Px \approx$ which objects are P?
- $s \models_g \forall x?Px \iff P$ has the same extension in all $w \in s$.

Mention-some questions

- $\overline{\exists} xPx \approx$ what is an instance of P?
- $s \models_g \overline{\exists} xPx \iff \exists d \in D$ s.t. $\forall w \in s : d \in P_w$

Expressing dependencies

Given any two questions Q, Q' expressible in the system, we can express the corresponding dependence by $Q \to Q'$. (Cf. the generality point above)

Examples

- $\forall x?Px \to \forall x?Qx$
  The extension of P determines the extension of Q.
- $\overline{\exists} xPx \to \overline{\exists} xQx$
  Any instance of P determines an instance of Q.
- $\exists xPx \to (\forall x?Px \to \overline{\exists} xQx)$
  Conditionally on the existence of a P, the extension of P determines an instance of Q.

Logical dependencies

Dependencies that hold in all states come out as valid entailments.
Examples:

- $\forall x(Px \leftrightarrow \neg Qx), \forall x?Px \models \forall x?Qx$
- $\exists xPx, \forall x?Px \models \overline{\exists} xPx$
- $\forall x?Px \models ?\exists xPx$

Logical dependencies

Dependencies that hold in all states come out as valid entailments.
Examples:

- $\forall x(Px \leftrightarrow \neg Qx), \forall x?Px \models \forall x?Qx$
- $\exists xPx, \forall x?Px \models \overline{\exists} xPx$
- $\forall x?Px \models ?\exists xPx$

A proof system for InqBQ would allow us to prove these dependencies.

Logical dependencies

Dependencies that hold in all states come out as valid entailments.
Examples:

- $\forall x(Px \leftrightarrow \neg Qx), \forall x?Px \models \forall x?Qx$
- $\exists xPx, \forall x?Px \models \overline{\exists} xPx$
- $\forall x?Px \models ?\exists xPx$

A proof system for InqBQ would allow us to prove these dependencies.

Proof system

A (candidate) natural deduction system for InqBQ consists of:

- standard intuitionistic i/e rules for each operator
- double negation for classical formulas
- a few specific inquisitive principles

Example

$\forall x(Px \leftrightarrow \neg Qx),\ \forall x?Px \models \forall x?Qx$

$$\dfrac{\dfrac{\forall x?Px}{?Py}\ \forall\mathsf{e} \qquad \dfrac{\dfrac{\dfrac{\forall x(Px \leftrightarrow \neg Qx)}{Py \leftrightarrow \neg Qy}\ \forall\mathsf{e} \qquad \dfrac{\dfrac{\dfrac{\forall x(Px \leftrightarrow \neg Qx)}{Py \leftrightarrow \neg Qy}\ \forall\mathsf{e}}{\neg Qy \rightarrow Py}\ \wedge\mathsf{e} \qquad [\neg Qy]}{Py}\ \rightarrow\mathsf{e} \qquad [\neg Py]}{\bot}\ \rightarrow\mathsf{i}}{\neg\neg Qy}\ \mathsf{DNE}}{Qy}\ \vee\mathsf{i}}{?Qy}\ \vee\mathsf{e}$$

$$\dfrac{\dfrac{\dfrac{\forall x?Px}{?Py}\ \forall\mathsf{e} \qquad \dfrac{\dfrac{\forall x(Px \leftrightarrow \neg Qx)}{Py \leftrightarrow \neg Qy}\ \forall\mathsf{e} \qquad [Py]}{Py \rightarrow \neg Qy}\ \wedge\mathsf{e}}{\neg Qy}\ \rightarrow\mathsf{e}}{?Qy}\ \vee\mathsf{i}}{\forall x?Qx}\ \forall\mathsf{i}$$

Question:
Is the candidate proof system complete?

Question:

Is the candidate proof system complete? OPEN

Question:
Is the candidate proof system complete? OPEN

Theorem (Grilletti, forthcoming)
The system is strongly complete for the classical antecedent fragment, i.e., the fragment where only classical formulas are allowed as antecedents of $\rightarrow$.

For instance, all logical dependencies above fall within the fragment, so we know they must be provable:

- $\forall x(Px \leftrightarrow \neg Qx), \forall x?Px \models \forall x?Qx$
- $\exists xPx, \forall x?Px \models \overline{\exists} xPx$
- $\forall x?Px \models ?\exists xPx$

Question (Axiomatizability)

Abstracting away from the particular proof system, we can ask:
is InqBQ axiomatizable in principle? Are its validities recursively enumerable?

Question (Axiomatizability)

Abstracting away from the particular proof system, we can ask:
is InqBQ axiomatizable in principle? Are its validities recursively enumerable?

Question (Compactness)

$\Phi \models \psi \stackrel{?}{\Longrightarrow} \exists$ finite $\Phi_0 \subseteq \Phi$ s.t. $\Phi_0 \models \psi$

Question (Axiomatizability)
Abstracting away from the particular proof system, we can ask:
is InqBQ axiomatizable in principle? Are its validities recursively enumerable?

Question (Compactness)
$\Phi \models \psi \stackrel{?}{\Longrightarrow} \exists$ finite $\Phi_0 \subseteq \Phi$ s.t. $\Phi_0 \models \psi$

Both are currently OPEN

Connection with powerset

The difficulties are brought about by $\rightarrow$, which quantifies over elements of $\wp(s) = \{t \mid t \subseteq s\}$.

$$\begin{aligned} s \models \varphi \rightarrow \psi \quad &\iff \quad \forall t \subseteq s : t \models \varphi \Rightarrow t \models \psi \\ &\iff \quad \forall t \in \wp(s) : t \models \varphi \Rightarrow t \models \psi \end{aligned}$$

Connection with powerset

The difficulties are brought about by $\rightarrow$, which quantifies over elements of $\wp(s) = \{t \mid t \subseteq s\}$.

$$\begin{aligned} s \models \varphi \rightarrow \psi \quad &\iff \quad \forall t \subseteq s : t \models \varphi \Rightarrow t \models \psi \\ &\iff \quad \forall t \in \wp(s) : t \models \varphi \Rightarrow t \models \psi \end{aligned}$$

Mathematical questions involving powersets are often tricky,
not just in set theory (continuum problem) but also in logic.

Example: Medvedev's logic

Consider the class of intuitionistic Kripke frames $\langle \wp(X) - \{\emptyset\}, \supseteq \rangle$ for X finite.
ML is the logic of this class. Is ML axiomatizable/r.e.?

Connection with powerset

The difficulties are brought about by $\rightarrow$, which quantifies over elements of $\wp(s) = \{t \mid t \subseteq s\}$.

$$\begin{aligned} s \models \varphi \rightarrow \psi \quad &\iff \quad \forall t \subseteq s : t \models \varphi \Rightarrow t \models \psi \\ &\iff \quad \forall t \in \wp(s) : t \models \varphi \Rightarrow t \models \psi \end{aligned}$$

Mathematical questions involving powersets are often tricky,
not just in set theory (continuum problem) but also in logic.

Example: Medvedev's logic

Consider the class of intuitionistic Kripke frames $\langle \wp(X) - \{\emptyset\}, \supseteq \rangle$ for X finite.
ML is the logic of this class. Is ML axiomatizable/r.e.? OPEN for >50 years.

Cardinality sentences

- To get a handle on the expressive power of InqBQ,
  it's interesting to look at what it can express about cardinalities.
- This is not interesting for statements ('flat' formulas)
  since then the expressive power coincides with that of FOL.
- But it becomes interesting in connection with cardinality questions.

Cardinality sentences

- To get a handle on the expressive power of InqBQ, it's interesting to look at what it can express about cardinalities.
- This is not interesting for statements ('flat' formulas) since then the expressive power coincides with that of FOL.
- But it becomes interesting in connection with cardinality questions.

Problem

Is the following expressible in InqBQ (possibly wrt finite models)?

(1) How many P are there?

That is, is there a sentence φ such that :

$$s \models \varphi \iff \#P_w \text{ is the same for all } w \in s$$

The inquisitive setting suggests a generalized notion of cardinality quantifier, which covers question-forming quantifiers like 'how many'. We can then ask:

Question

what cardinality quantifiers are expressible in InqBQ?

The inquisitive setting suggests a generalized notion of cardinality quantifier, which covers question-forming quantifiers like 'how many'. We can then ask:

Question

what cardinality quantifiers are expressible in InqBQ?

Using a suitably adapted version of EF games, we can answer this question.

Theorem (Grilletti and C., forthcoming)

A cardinality quantifier is expressible in InqBQ iff it is downward closed and, for some $n \in \mathbb{N}$, it does not distinguish cardinalities above n.

The inquisitive setting suggests a generalized notion of cardinality quantifier, which covers question-forming quantifiers like 'how many'. We can then ask:

Question
what cardinality quantifiers are expressible in InqBQ?

Using a suitably adapted version of EF games, we can answer this question.

Theorem (Grilletti and C., forthcoming)
A cardinality quantifier is expressible in InqBQ iff it is downward closed and, for some $n \in \mathbb{N}$, it does not distinguish cardinalities above n.

Corollary
The question 'how many objects are P' is not expressible,
even in restriction to finite models.

Interesting contrast

- What objects are P? $\rightsquigarrow$ expressible
- How many objects are P? $\rightsquigarrow$ not expressible

Interesting contrast

- What objects are P? $\rightsquigarrow$ expressible
- How many objects are P? $\rightsquigarrow$ not expressible

That is, we can express that the extension of P is constant, but not that the cardinality of this extension is constant.

Interesting contrast

- What objects are P? $\rightsquigarrow$ expressible
- How many objects are P? $\rightsquigarrow$ not expressible

That is, we can express that the extension of P is constant,
but not that the cardinality of this extension is constant.

In view of the importance of 'how many' questions, this suggests a natural extension of InqBQ with the 'how many' quantifier H:

$$s \models Hx.\alpha(x) \iff \text{cardinality of extension of } \alpha(x) \text{ is constant in } s$$

Interesting contrast

- What objects are P? $\rightsquigarrow$ expressible
- How many objects are P? $\rightsquigarrow$ not expressible

That is, we can express that the extension of P is constant,
but not that the cardinality of this extension is constant.

In view of the importance of 'how many' questions, this suggests a natural extension of InqBQ with the 'how many' quantifier H:

$$s \models Hx.\alpha(x) \iff \text{cardinality of extension of } \alpha(x) \text{ is constant in } s$$

Task

Study the meta-theoretic properties of the resulting extension InqBQ+H.


THANK
YOU
GRACIAS
ARIGATO
SHUKURIA
GOZAIMASHITA
EFCHARISTO
DANKSCHEEN
JUSPAXAR
KOMAPSUMNIDA
MAAKE
GRAZIE
MEHRBANI
PALDIES
SUKSAMA
EKHMET
TASHAKKUR ATU
YAQHANYELAY
TINGKI
BİYAN
SHUKRIA
BOLZİN
MERCI