Structural Logical Relations with Case Analysis and Equality Reasoning

Ulrik Rasmussen Andrzej Filinski

Department of Computer Science
University of Copenhagen

LFMTP, Boston, MA
September 23, 2013

Motivation

- Logical relations (LR) are a powerful proof technique, but difficult to formalize in Twelf and similar systems.
- Method to do so (*structural logical relations*) devised by [Schürmann and Sarnat, 2008]: Formalizes weak normalization and completeness of equivalence checking for simply typed λ-calculus.
- Minimal, pure λ-calculus.
- Can we use this for "real" programming languages?

Our Contributions

- Extension of structural logical relations allowing more proofs to be formalized.
- Further insight into the structure of logical-relations based proofs.
- Demonstration of proofs of observational equivalence.
- In this talk: High-level perspective; see paper for technical details.

Example 1: Termination

Definition (λ^{nat})

Naturals	n	::	Nat	::=	$\mathsf{z} \mid \mathsf{s}\ n$
Expressions	e, v	::	Exp	::=	$x \mid \mathsf{lam}\ x.\, e_0 \mid \mathsf{app}\ e_1\ e_2 \mid \mathsf{num}\ n$
Types	τ	::	Tp	::=	$\mathsf{nat} \mid \mathsf{arr}\ \tau_2\ \tau_0$
CBN Eval.	$\mathcal{E}$	::	$\boxed{e \Downarrow v}$		
Typing	$\mathcal{T}$	::	$\boxed{x_1 : \tau_1, \ldots, x_n : \tau_n \triangleright e : \tau}$		

Theorem (Termination)

For any e where $\triangleright\ e : \mathsf{nat}$, there exists a v such that $e \Downarrow v$.

Example 1: Logical Relation

- Termination proof requires a logical relation:

Definition (Logical Relation for Termination)

$$
\begin{aligned}
e \in [\![\mathsf{nat}]\!] &\iff \exists n.\, e \Downarrow \mathsf{num}\ n \\
e \in [\![\mathsf{arr}\ \tau_2\ \tau_0]\!] &\iff \forall e_2.\, e_2 \in [\![\tau_2]\!] \supset \mathsf{app}\ e\ e_2 \in [\![\tau_0]\!]
\end{aligned}
$$

- Extend to open expressions: For $\Gamma = x_1 : \tau_1, \ldots, x_n : \tau_n$:

$$
\Gamma \vdash e \in [\![\tau]\!] \iff \begin{array}{l} \forall e_1 \in [\![\tau_1]\!] \cdots e_n \in [\![\tau_n]\!]. \\ \quad e[e_1 \cdots e_n / x_1 \cdots x_n] \in [\![\tau]\!] \end{array}
$$

- Fundamental Theorem: If $\Gamma \triangleright e : \tau$ then $\Gamma \vdash e \in [\![\tau]\!]$.

- Representing LR at arrow types problematic. Twelf only supports metatheorems on $\forall\exists$-form.

Use an Assertion Logic

Structural Logical Relations [Schürmann and Sarnat, 2008]:

Definition (Assertion Logic, $\stackrel{\text{eval}}{\Longrightarrow}$)

Propositions:	A, B	::	Form	::=	$\forall^{\text{Exp}}\alpha.\, A \mid \exists^{\text{Nat}}\alpha.\, A$
				$\mid$	$A \supset B$
				$\mid$	$\textbf{eval}(e, v)$
Assumptions:	Δ	::	Assm	::=	$\{A_1, \ldots, A_n\}$ (Unordered)
Parameters:	Ξ	::	Ctx	::=	$\cdot \mid \Xi, \alpha : \text{Nat} \mid \Xi, \alpha : \text{Exp}$

"Cut-full" sequent: $\boxed{\Xi \mid \Delta \stackrel{\bullet}{\Longrightarrow} A}$

"Cut-free" sequent: $\boxed{\Xi \mid \Delta \stackrel{\circ}{\Longrightarrow} A}$

- $\stackrel{\circ}{\Longrightarrow}$ $\textbf{eval}(\_, \_)$ axiomatizes $\_ \Downarrow \_$:

Theorem (Extraction)

If $\cdot \mid \emptyset \stackrel{\circ}{\Longrightarrow} \textbf{eval}(e, v)$, *then* $e \Downarrow v$.

Fundamental Theorem

- LR representation: Map types to propositions w/bound expression:

$$[\![\tau]\!] :: \mathsf{Exp} \to \mathsf{Form}$$

Definition (Logical Relation for Termination, Assertion-Level)

$$\begin{array}{rcl} [\![\mathsf{nat}]\!](e) & \Longleftrightarrow & \exists^{\mathsf{Nat}} n.\, \mathbf{eval}(e, \mathsf{num}\; n) \\ [\![\mathsf{arr}\; \tau_2\; \tau_0]\!](e) & \Longleftrightarrow & \forall^{\mathsf{Exp}} e_2.\, [\![\tau_2]\!](e_2) \supset [\![\tau_0]\!](\mathsf{app}\; e\; e_2) \end{array}$$

Theorem (Fundamental Theorem)

For any e, if

$$x_1 : \tau_1, \ldots, x_n : \tau_n \triangleright e : \tau,$$

then

$$x_1 : \mathsf{Exp}, \ldots, x_n : \mathsf{Exp} \mid [\![\tau_1]\!](x_1), \ldots, [\![\tau_n]\!](x_n) \stackrel{\bullet}{\Longrightarrow} [\![\tau]\!](e).$$

- Note: Induction lives entirely on the meta-level!

Cut Elimination

- Corollary: $\triangleright\ e : \text{nat}$ implies $\cdot | \emptyset \xrightarrow{\bullet} \exists v.\, \mathbf{eval}(e, v)$.

- By *extraction*, termination reduced to proving *cut elimination*:

Theorem (Cut Elimination)

If $\Xi | \Delta \xrightarrow{\bullet} A$, *then* $\Xi | \Delta \xrightarrow{\circ} A$

- In Twelf: Syntactic proof due to [Pfenning, 2000]. Bulk of work in:

Lemma (Cut Admissibility)

If $\Xi | \Delta \xrightarrow{\circ} A$ *and* $\Xi | \Delta, A \xrightarrow{\circ} C$ *then* $\Xi | \Delta \xrightarrow{\circ} C$.

Extending to More Expressive Languages

- Languages just slightly more expressive than simply typed λ-calculus require stronger assertion logic.
- Specifically, equality reasoning and case-analysis principles.
- Assertion logic can only be strengthened if it retains cut-admissibility.

Example 2: λ-calculus + ifz

Definition ($\lambda^{\text{nat,ifz}}$)

Naturals	n	::	Nat	::=	$\mathsf{z} \mid \mathsf{s}\ n$
Expressions	e	::	Exp	::=	$x \mid \mathsf{lam}\ x.\ e_0 \mid \mathsf{app}\ e_1\ e_2 \mid \mathsf{num}\ n$
				$\mid$	$\mathsf{ifz}(e_0, e_1, e_2)$
Types	τ	::	Tp	::=	$\mathsf{nat} \mid \mathsf{arr}\ \tau_2\ \tau_0$
CBN Eval.	$\mathcal{E}$	::	$\boxed{e \Downarrow v}$		
Typing	$\mathcal{T}$	::	$\boxed{\Gamma \triangleright e : \tau}$		

- Fund thm.: By IH, get $[\![\mathsf{nat}]\!](e_0) \equiv \exists^{\mathsf{Nat}} n.\ \mathbf{eval}(e_0, \mathsf{num}\ n)$. Select one of branches e_1 or e_2 based on n.
- Structure of terms opaque to assertion logic.
- Specify structure explicitly in LR.

Example 2: Logical Relation, Assertion Logic

Definition (Assertion Logic ($\Longrightarrow^{\text{eval,eq}}$))

Propositions:	A, B	::	Form	::=	$\forall^{\text{Exp}}\alpha.\, A \mid \exists^{\text{Nat}}\alpha.\, A$
				$\mid$	$A \supset B \mid A \wedge B \mid A \vee B$
				$\mid$	$\mathbf{eval}(e, v) \mid \mathbf{eq}(n, n')$
Assumptions:	Δ	::	Assm	::=	$\{A_1, \ldots, A_n\}$ (Unordered)
Parameters:	Ξ	::	Ctx	::=	$\cdot \mid \Xi, \alpha : \text{Nat} \mid \Xi, \alpha : \text{Exp}$

Proof sequent: $\boxed{\Xi \middle| \Delta \xrightarrow{c} A}$ $(c \in \{\bullet, \circ\})$

Definition (Logical Relation for Termination, Assertion-Level)

$$\llbracket \text{nat} \rrbracket(e) \iff \exists^{\text{Nat}} n.\, \mathbf{eval}(e, \text{num}\ n) \wedge (\mathbf{eq}(n, \text{z}) \vee \exists^{\text{Nat}} n'.\, \mathbf{eq}(n, \text{s}\ n'))$$

$$\llbracket \text{arr}\ \tau_2\ \tau_0 \rrbracket(e) \iff \forall^{\text{Exp}} e_2.\, \llbracket \tau_2 \rrbracket(e_2) \supset \llbracket \tau_0 \rrbracket(\text{app}\ e\ e_2)$$

Equality

- $\mathbf{eq}(n, n')$ axiomatizes syntactic equality:

$$\frac{}{\Xi \middle| \Delta \xrightarrow{c} \mathbf{eq}(n, n)}$$

- Cannot show cut-elim for logic w/general equality conversion.

- Must restrict equality reasoning to *leaves* of proofs, i.e., atomic formulas:

$$\frac{\Xi \middle| \Delta \xrightarrow{c} \mathbf{eval}(e[n_1/x_1], v[n_2/x_2]) \quad \begin{array}{c} \Xi \middle| \Delta \xrightarrow{c} \mathbf{eq}(n_1, n_1') \\ \Xi \middle| \Delta \xrightarrow{c} \mathbf{eq}(n_2, n_2') \end{array}}{\Xi \middle| \Delta \xrightarrow{c} \mathbf{eval}(e[n_1'/x_1], v[n_2'/x_2])}$$

Example 3: λ-calculus + case

Definition ($\lambda^{\text{nat,case}}$)

Naturals	n	::	Nat	::=	$\mathsf{z} \mid \mathsf{s}\ n$
Expressions	e	::	Exp	::=	$x \mid \mathsf{lam}\ x.\ e_0 \mid \mathsf{app}\ e_1\ e_2 \mid \mathsf{num}\ n$
				$\mid$	$\mathsf{case}(e_0, e_1, x.\ e_2)$
Types	τ	::	Tp	::=	$\mathsf{nat} \mid \mathsf{arr}\ \tau_2\ \tau_0$
CBN Eval.	$\mathcal{E}$	::	$\boxed{e \Downarrow v}$		
Typing	$\mathcal{T}$	::	$\boxed{\Gamma \triangleright e : \tau}$		

- Still need to select branch based on $[\![\mathsf{nat}]\!](e_0)$.
- In subcase where $\Delta \mid \Xi \overset{\bullet}{\Longrightarrow} \mathbf{eval}(e_0, \mathsf{num}\ (\mathsf{s}\ n'))$: By IH, get $\Xi, x : \mathsf{Exp} \mid \Delta, [\![\mathsf{nat}]\!](x) \overset{\bullet}{\Longrightarrow} [\![\tau]\!](e_2)$. *Instantiate* LR for $e_2[\mathsf{num}\ n'/x]$: Need to show $\overset{\bullet}{\Longrightarrow} [\![\mathsf{nat}]\!](\mathsf{num}\ n')$.

Example 3: Logical Relation, Assertion Logic

Definition (Assertion Logic ($\xRightarrow{\text{eval,eq}}$))

Propositions: $A, B \;::\; \textsf{Form} \;::=\; \forall^{\textsf{Exp}}\alpha.\, A \mid \exists^{\textsf{Nat}}\alpha.\, A$

$\mid\; A \supset B \mid A \wedge B \mid A \vee B$

$\mid\; \mathbf{eval}(e, v) \mid \mathbf{eq}(n, n')$

Assumptions: $\Delta \;::\; \textsf{Assm} \;::=\; \{A_1, \ldots, A_n\}$ (Unordered)

Parameters: $\Xi \;::\; \textsf{Ctx} \;::=\; \cdot \mid \Xi, \alpha : \textsf{Nat} \mid \Xi, \alpha : \textsf{Exp}$

Proof sequent: $\boxed{\Xi | \Delta \xrightarrow{c} A}$ $(c \in \{\bullet, \circ\})$

Definition (Logical Relation for Termination, Assertion-Level)

$$[\![\textsf{nat}]\!](e) \iff \exists^{\textsf{Nat}} n.\, \mathbf{eval}(e, \textsf{num}\; n)$$

$$\wedge\, (\mathbf{eq}(n, \textsf{z}) \vee (\exists^{\textsf{Nat}} n'.\, \mathbf{eq}(n, \textsf{s}\; n')$$

$$\wedge\, (\mathbf{eq}(n', \textsf{z}) \vee \exists^{\textsf{Nat}} n''.\, \cdots)))$$

$$[\![\textsf{arr}\; \tau_2\; \tau_0]\!](e) \iff \forall^{\textsf{Exp}} e_2.\, [\![\tau_2]\!](e_2) \supset [\![\tau_0]\!](\textsf{app}\; e\; e_2)$$

Example 3: Logical Relation, Assertion Logic

Definition (Assertion Logic ($\Longrightarrow^{\text{eval},\text{eq},\text{nat}^+}$))

Propositions:	A, B	$::$	Form	$::=$	$\forall^{\text{Exp}}\alpha.\, A \mid \exists^{\text{Nat}}\alpha.\, A$
				$\mid$	$A \supset B \mid A \wedge B \mid A \vee B$
				$\mid$	$\mathbf{eval}(e, v) \mid \mathbf{eq}(n, n') \mid \mathbf{nat}^+(n)$
Assumptions:	Δ	$::$	Assm	$::=$	$\{A_1, \ldots, A_n\}$ (Unordered)
Parameters:	Ξ	$::$	Ctx	$::=$	$\cdot \mid \Xi, \alpha : \text{Nat} \mid \Xi, \alpha : \text{Exp}$

Proof sequent: $\boxed{\Xi \mid \Delta \overset{c}{\Longrightarrow} A}$ $(c \in \{\bullet, \circ\})$

Definition (Logical Relation for Termination, Assertion-Level)

$$[\![\text{nat}]\!](e) \iff \exists^{\text{Nat}} n.\, \mathbf{eval}(e, \text{num}\ n) \wedge \mathbf{nat}^+(n)$$

$$[\![\text{arr}\ \tau_2\ \tau_0]\!](e) \iff \forall^{\text{Exp}} e_2.\, [\![\tau_2]\!](e_2) \supset [\![\tau_0]\!](\text{app}\ e\ e_2)$$

Assertion Logic With Case-Analysis on Naturals

$$\frac{}{\Xi \,|\, \Delta \overset{c}{\Longrightarrow} \mathbf{nat}^{+}(\mathsf{z})} \qquad \frac{\Xi \,|\, \Delta \overset{c}{\Longrightarrow} \mathbf{nat}^{+}(n)}{\Xi \,|\, \Delta \overset{c}{\Longrightarrow} \mathbf{nat}^{+}(\mathsf{s}\ n)}$$

$$\frac{\Xi \,|\, \Delta, \mathbf{eq}(n, \mathsf{z}) \overset{c}{\Longrightarrow} C \qquad \Xi, n' : \mathsf{Nat} \,|\, \Delta, \mathbf{eq}(n, \mathsf{s}\ n'), \mathbf{nat}^{+}(n') \overset{c}{\Longrightarrow} C}{\Xi \,|\, \Delta, \mathbf{nat}^{+}(n) \overset{c}{\Longrightarrow} C}$$

- $\mathbf{nat}^{+}(n)$ proof: *structural witness* for some n.
- As-is, Pfenning's cut-admissibility proof does not work for logic with left-rules on atomic propositions.
- Can be made to work as long as an index term always gets smaller in subderivations. For $\mathbf{nat}^{+}(n)$: n gets smaller.

Case-Analysis on *Derivations*?

- Required in, e.g., proofs of observational equivalence (see paper).
- Observation: For **eval**(e, v), indices do not get smaller in sub-proofs. To be able to add left-rule, index by explicit metric, e.g.: **eval**(e, v, d).
- Alternatively: Treat object-language derivations as *terms* with *dependent sorts*.
- In the following: Will show minimal example.

Example 4: λ-calculus + case + numeral constructors

Definition ($\lambda^{\mathrm{sz,case}}$)

$$
\begin{array}{llcll}
\text{Expressions} & e & :: & \mathsf{Exp} & ::= \; x \mid \mathsf{lam}\, x.\, e_0 \mid \mathsf{app}\, e_1\, e_2 \mid \mathsf{z} \mid \mathsf{s}\, e_0 \\
 & & & & \quad\mid \; \mathsf{case}(e_0, e_1, x.\, e_2) \\
\text{Types} & \tau & :: & \mathsf{Tp} & ::= \; \mathsf{nat} \mid \mathsf{arr}\, \tau_2\, \tau_0 \\
\text{CBN Eval.} & \mathcal{E} & :: & \boxed{e \Downarrow v} & \\
\text{Typing} & \mathcal{T} & :: & \boxed{\Gamma \triangleright e : \tau} & \\
\text{Num} & \mathcal{N} & :: & \boxed{v\ \#} &
\end{array}
$$

- Numerals characterized in *object-language judgment*:

$$
\frac{}{\mathsf{z}\ \#} \qquad \frac{v\ \#}{\mathsf{s}\, v\ \#}
$$

- *Could* axiomatize as atomic formula, $A ::= \ldots \mid \mathbf{isnum}(v)$.

- Alternatively: Treat $\boxed{v\ \#}$ as a *dependent sort*; add structural witness formula.

Example 4: Logical Relation, Assertion Logic

Definition (Assertion Logic ($\Longrightarrow_{\Pi}^{\mathsf{eval},\mathsf{eq},\mathsf{num}^+}$))

Propositions:	A, B	::	Form	::=	$\forall^{\mathsf{Exp}}\alpha.\, A \mid \exists^{\mathsf{Exp}}\alpha.\, A \mid \exists^{(e\ \#)}\alpha.\, A$
				$\mid$	$A \supset B \mid A \wedge B \mid A \vee B$
				$\mid$	$\mathbf{eval}(e, v) \mid \mathbf{eq}(e, e')$
				$\mid$	$\mathbf{num}^+(\mathcal{N})$
Assumptions:	Δ	::	Assm	::=	$\{A_1, \ldots, A_n\}$ (Unordered)
Parameters:	Ξ	::	Ctx	::=	$\cdot \mid \Xi, \alpha : \mathsf{Exp} \mid \Xi, \alpha : (e\ \#)$
Proof sequent:			$\boxed{\Xi \mid \Delta \xrightarrow{c} A}$		$(c \in \{\bullet, \circ\})$

Definition (Logical Relation for Termination, Assertion-Level)

$$\llbracket \mathsf{nat} \rrbracket(e) \iff \exists^{\mathsf{Exp}} v.\, \mathbf{eval}(e, v) \wedge \exists^{(v\ \#)}\mathcal{N}.\, \mathbf{num}^+(\mathcal{N})$$

$$\llbracket \mathsf{arr}\ \tau_2\ \tau_0 \rrbracket(e) \iff \forall^{\mathsf{Exp}} e_2.\, \llbracket \tau_2 \rrbracket(e_2) \supset \llbracket \tau_0 \rrbracket(\mathsf{app}\ e\ e_2)$$

Cut-Elimination for Logic with Dependent Sorts

- "Well-sortedness" must be compositional w.r.t. substitution:

Theorem (Compositionality)

If $o :: S$ *and* $\Xi_1, \alpha : S, \Xi_2 \mid \Delta \xrightarrow{c} A$ *then* $\Xi_1, \Xi_2[o/\alpha] \mid \Delta[o/\alpha] \xrightarrow{c} A[o/\alpha]$.

- "Free" theorem: everything is represented in LF, contexts Ξ in particular.

- Pfenning's cut-admissibility theorem requires no changes!

Equality and Case-Analysis

- Need to take care if we want to add equality conversion axioms to judgments on which we reason by case distinction.

- **Example**: Let $\boxed{e \overset{\star}{=} e'}$ be axiomatization of syntactic equality. Treat as sort.

$$\frac{}{e \overset{\star}{=} e} \qquad \frac{e \overset{\star}{=} e'}{\mathsf{s}\ e \overset{\star}{=} \mathsf{s}\ e'} \qquad \frac{\mathsf{s}\ e \overset{\star}{=} \mathsf{s}\ e'}{e \overset{\star}{=} e'} \qquad \frac{\mathsf{s}\ e_0 \overset{\star}{=} \mathsf{z}}{e \overset{\star}{=} e'} \qquad \frac{e \overset{\star}{=} e' \quad e' \overset{\star}{=} e''}{e \overset{\star}{=} e''} \qquad \cdots$$

  - **Goal:** From $\mathsf{s}\ n \overset{\star}{=} \mathsf{s}\ n'$ and $n\ \#$, infer $n'\ \#$.

- Quantify over alternative judgment $\boxed{e\ \#^{=}}$ equivalent to $\boxed{e\ \#}$, but with explicit equality rules.

Resulting Assertion Logic

Definition (Assertion Logic $\left(\Longrightarrow^{\text{eval},\text{num}^+}_{\Pi,\ \#^=,\overset{\star}{=}}\right)$)

Propositions:	A, B	::	Form	$::=$	$\forall^{\mathsf{Exp}}\alpha.\,A \mid \exists^{\mathsf{Exp}}\alpha.\,A$
				$\mid$	$\exists^{(e\ \#^=)}\alpha.\,A \mid \exists^{(e \overset{\star}{=} e')}\alpha.\,A$
				$\mid$	$A \supset B \mid A \wedge B \mid A \vee B$
				$\mid$	$\mathbf{eval}(e, v) \mid \mathbf{num}^+(\mathcal{N})$
Assumptions:	Δ	::	Assm	$::=$	$\{A_1, \ldots, A_n\}$ (Unordered)
Parameters:	Ξ	::	Ctx	$::=$	$\cdot \mid \Xi, \alpha : \mathsf{Exp} \mid \Xi, \alpha : (e\ \#^=)$
				$\mid$	$\Xi, \alpha : (e \overset{\star}{=} e')$

Proof sequent: $\boxed{\Xi \mid \Delta \overset{c}{\Longrightarrow} A}$ $(c \in \{\bullet, \circ\})$

Retain Canonicity of Derivations

- How to define rules for $\boxed{e\ \#^{=}}$?

- **Bad:** Add extra rule $\Rightarrow$ extra case to handle in all proofs:

$$\frac{}{\mathsf{z}\ \#^{=}}\,\mathrm{nz} \qquad \frac{v'\ \#^{=}}{\mathsf{s}\ v'\ \#^{=}}\,\mathrm{ns} \qquad \frac{v\ \#^{=} \quad v \stackrel{\star}{=} v'}{v'\ \#^{=}}\,\mathrm{conv}$$

- **Good:** Make equality intrinsic property of all rules:

$$\frac{v \stackrel{\star}{=} \mathsf{z}}{v\ \#^{=}}\,\mathrm{nz}' \qquad \frac{v'\ \#^{=} \quad v \stackrel{\star}{=} \mathsf{s}\ v'}{v\ \#^{=}}\,\mathrm{ns}'$$

- Derivations still canonical. Conversions pushed to equality derivations.

Example

- **Given** $\mathcal{Q} :: \mathsf{s}\, e' \stackrel{\star}{=} \mathsf{s}\, e$ **and** $\mathcal{N} :: e\ \#^{=}$, **show** $e'\ \#^{=}$.

- $\mathcal{N}$ must end in nz$'$ or ns$'$.

- Case $\mathcal{N} = \dfrac{\overset{\mathcal{N}'}{e_0\ \#^{=}} \qquad \overset{\mathcal{Q}'}{e \stackrel{\star}{=} \mathsf{s}\, e_0}}{e\ \#^{=}}\,\mathrm{ns}'$ (case for nz$'$ analogous).

- Obtain result by

$$\dfrac{\overset{\mathcal{N}'}{e_0\ \#^{=}} \qquad \dfrac{\dfrac{\overset{\mathcal{Q}}{\mathsf{s}\, e' \stackrel{\star}{=} \mathsf{s}\, e}}{e' \stackrel{\star}{=} e} \qquad \overset{\mathcal{Q}'}{e \stackrel{\star}{=} \mathsf{s}\, e_0}}{e' \stackrel{\star}{=} \mathsf{s}\, e_0}}{e'\ \#^{=}}\,\mathrm{ns}'$$

Summary

- **Results**
  - Extension of SLR method to allow reasoning by case-analysis and equality.
  - More proofs can be formalized: see paper for observational equivalence proofs.
  - Nice property: Pfenning's cut-elim proof works for dependentently-sorted logic.

Summary

- **Results**
  - Extension of SLR method to allow reasoning by case-analysis and equality.
  - More proofs can be formalized: see paper for observational equivalence proofs.
  - Nice property: Pfenning's cut-elim proof works for dependentently-sorted logic.
- **Future work**
  - Lots of boilerplate. Code generation or extension of Twelf?
  - Experiment with stronger logics – no termination guarantees for cut-elim though.

Summary

- **Results**
  - Extension of SLR method to allow reasoning by case-analysis and equality.
  - More proofs can be formalized: see paper for observational equivalence proofs.
  - Nice property: Pfenning's cut-elim proof works for dependentently-sorted logic.
- **Future work**
  - Lots of boilerplate. Code generation or extension of Twelf?
  - Experiment with stronger logics – no termination guarantees for cut-elim though.
- **Questions?**
  - Code, paper, slides: see `http://www.utr.dk/`.

Related Work I

Frank Pfenning.
Structural cut elimination: I. Intuitionistic and classical logic.
Information and Computation, 157(1–2):84–141, 2000.

Carsten Schürmann and Jeffrey Sarnat.
Structural logical relations.
In *Proceedings of the 2008 23rd Annual IEEE Symposium on Logic in Computer Science*, LICS '08, pages 69–80, Washington, DC, USA, 2008. IEEE Computer Society.