Towards Dualized Type Theory

Aaron Stump, Harley Eades III, Ryan McCleeary
Dept. of Computer Science
The University of Iowa
Iowa City, Iowa, USA
Duality in Logic

- Basic duality in logic between truth and falsehood
- Some propositional connectives are duals
  - \( \top \) and \( \bot \)
  - \( T \land T' \) and \( T \lor T' \)
- Duality clear in sequent calculus:

  \[
  \Gamma \vdash \top, \Delta \\
  \Gamma \vdash T, \Delta \quad \Gamma \vdash T', \Delta \\
  \quad \Gamma \vdash T \land T', \Delta \\
  \quad \Gamma, \bot \vdash \Delta \\
  \quad \Gamma, T \vdash \Delta \quad \Gamma, T' \vdash \Delta \\
  \quad \Gamma, T \lor T' \vdash \Delta \\
  \quad \Gamma, T_i \vdash \Delta \quad i \in \{1, 2\} \\
  \quad \Gamma \vdash T_1 \land T_2 \vdash \Delta \\
  \quad \Gamma \vdash T_1 \lor T_2 \vdash \Delta
  \]

- These dualities hold in intuitionistic, classical logic
Duality in programming/type theory

- Basic duality between input (+) and output (−)
- But the duality is very poorly explored
  - Input variables
  - Not really output variables, except maybe continuations
  - Positive term constructs like pairs, but
  - No negative term constructs

Computational classical type theories

- $\lambda\mu$-calculus, $\lambda\Delta$-calculus, $\bar{\lambda}\mu\bar{\mu}$-calculus, Dual Calculus
- Duality is central
- Control operators ($\mu x.p \bullet n$)
- But due to control operator, no canonicity:

\[
\mu x.(\text{in}_2 \lambda y.\mu x'.(\text{in}_1 y) \bullet x) \bullet x : A \lor \neg A
\]
Duality in programming/type theory

- Basic duality between input (+) and output (−)
- But the duality is very poorly explored
  - Input variables
  - Not really output variables, except maybe continuations
  - Positive term constructs like pairs, but
  - No negative term constructs

Computational classical type theories

- $\lambda\mu$-calculus, $\lambda\Delta$-calculus, $\bar{\lambda}\mu\bar{\mu}$-calculus, Dual Calculus
- Duality is central
- Control operators ($\mu x.p \bullet n$)
- But due to control operator, no canonicity:

$$\mu x.(in_2 \lambda y.\mu x'.(in_1 y) \bullet x) \bullet x : A \lor \neg A$$

Goal: import constructive duality from logic to programming/type theory
But first: constructive duality!

- What is the constructive dual of implication?
- Constructive implication is modal; so is its dual
- Known as “subtraction” (“exclusion”, “pseudo-difference”)
- A modest line of work starting with Rauszer (1970s)
- Tricky:
  - Crolard (TCS 2001) gives proof system
  - Sound and complete, cut elimination conjectured
  - Counterexamples found later (Pinto and Uustalu 2009)
  - Correct cut-free systems (Pinto and Uustalu 2009, Goré et al. 2007)
  - But no type theories yet
Dualized Intuitionistic Logic (DIL)

- Incorporate duality into the syntax

\[
polarities \quad p \quad ::= \quad + \mid - \\
formulas \quad T \quad ::= \quad A \mid \langle p \rangle \mid T \land_p T' \mid T \to_p T'
\]

- \(\langle + \rangle\) is \(\top\)
- \(\langle - \rangle\) is \(\bot\)
- \(A \land_+ B\) is \(A \land B\)
- \(A \land_- B\) is \(A \lor B\)
- \(A \to_+ B\) is \(A \to B\)
- \(A \to_- B\) is \(B \to A\)
Semantics of formulas

- Kripke models \((W, \preceq, V)\)
  - \(W\) is a non-empty set of worlds
  - \(\preceq\) is a reflexive, transitive relation on \(W\)
  - \(V(w)\) is set of atoms \(A\) true in world \(w\)
  - Require: \(w \preceq w' \implies V(w) \subseteq V(w')\).

- Semantics:

\[
\begin{align*}
\llbracket A \rrbracket_w &\iff A \in V(w) \\
\llbracket \langle + \rangle \rrbracket_w &\iff \text{true} \\
\llbracket \langle - \rangle \rrbracket_w &\iff \text{false} \\
\llbracket T \land_+ T' \rrbracket_w &\iff \llbracket T \rrbracket_w \land \llbracket T' \rrbracket_w \\
\llbracket T \land_- T' \rrbracket_w &\iff \llbracket T \rrbracket_w \lor \llbracket T' \rrbracket_w \\
\llbracket T \rightarrow_+ T' \rrbracket_w &\iff \forall w'. w \preceq w' \implies \llbracket T \rrbracket_{w'} \implies \llbracket T' \rrbracket_{w'} \\
\llbracket T \rightarrow_- T' \rrbracket_w &\iff \exists w'. w \preceq w' \land \lnot \llbracket T \rrbracket_{w'} \land \llbracket T' \rrbracket_{w'}
\end{align*}
\]

- Key fact: \(\lnot \llbracket T \rightarrow_- T' \rrbracket_w \iff \forall w'. w \preceq w' \implies \lnot \llbracket T \rrbracket_{w'} \implies \lnot \llbracket T' \rrbracket_{w'}\)

- Monotonicity Theorem: \(w \preceq w'\) and \(\llbracket T \rrbracket_w\) implies \(\llbracket T \rrbracket_{w'}\)
Kripke models, classically

Define \( \sim T := T \rightarrow_{-} \langle + \rangle \)

\([T \land_{-} \sim T]_{w} = true\)

“Either \(T\) is true now, or there is an earlier world where it is false”

\([\sim \sim T \rightarrow_{+} T]_{w} = true\)

“For any future world \(w\), if there is an earlier world (\(w'\)) where it is not the case that \(T\) is false in a previous world, then \(T\) is true in \(w\)”
Kripke models, classically

Define $\sim T := T \rightarrow_\neg \langle+\rangle$

$\llbracket T \land \sim T \rrbracket_w = true$

“Either $T$ is true now, or there is an earlier world where it is false”

$\llbracket \sim \sim T \rightarrow_+ T \rrbracket_w = true$

“For any future world $w$, if there is an earlier world ($w'$) where it is not the case that $T$ is false in a previous world, then $T$ is true in $w$”

Can lose canonicity with opposite polarity modality ($\rightarrow_\bar{p}$)
Devising a Proof System

- Start with labeled sequent calculus of Pinto, Uustalu, TABLEAUX 2009.

  \[
  \text{world names (labels) } n \\
  \text{contexts } \Gamma ::= \cdot | \Gamma, p \ T@n
  \]

- Treat \( \Gamma \) as a set
- Finite graphs \( G \) on world names
- Pinto, Uustalu’s judgments: \( \Gamma \vdash_G \Delta \) (unsigned \( \Gamma, \Delta \))
- We use instead \( G; \Gamma \vdash^p_n T \)
- Intended semantics:
  - For any Kripke model \( K \)
  - whose graph structure satisfies \( G \)
  - and where \( p \ T@n' \in \Gamma \) implies that in world corresponding to \( n' \), \( pT \) holds
  - then in the world \( w \) corresponding to \( n \),
  - \( p \llbracket T \rrbracket_w \) holds
Proof rules

\[
\frac{G \vdash n \preceq_p n'}{G; \Gamma, p \mathrel{\top} n \vdash_p^{n'} T} \quad \text{AX}
\]

\[
\frac{n' \not\in |G|}{(G, n \preceq_p n'); \Gamma, p \mathrel{\top} n' \vdash_p^{n'} T_2}{G; \Gamma \vdash_p^{n} T_1 \to p \mathrel{\top} T_2} \quad \text{IMP}
\]

\[
\frac{G; \Gamma \vdash^{n} T_1 \quad G; \Gamma \vdash^{n} T_2}{G; \Gamma \vdash^{n} T_1 \wedge_p T_2} \quad \text{AND}
\]

\[
\frac{G; \Gamma \vdash^{n} T_2}{G; \Gamma \vdash^{n} T_1 \wedge \bar{p} \mathrel{\top} T_2} \quad \text{ANDBAR2}
\]

\[
\frac{G; \Gamma, \bar{p} \mathrel{\top} n \vdash_p^{n'} T' \quad G; \Gamma, \bar{p} \mathrel{\top} n \vdash^{n'} \bar{p} \mathrel{\top} T'}{G; \Gamma \vdash^{n} T} \quad \text{CUT}
\]

\[
\frac{G \vdash n \preceq\bar{p}^* n'}{G; \Gamma \vdash_{n} \langle p \rangle} \quad \text{UNIT}
\]

\[
\frac{G \vdash n \preceq\bar{p}^* n'}{G; \Gamma \vdash_{n'} T_1 \quad G; \Gamma \vdash_{n'} T_2}{G; \Gamma \vdash_{n} T_1 \to \bar{p} \mathrel{\top} T_2} \quad \text{IMPBAR}
\]

\[
\frac{G \vdash n \preceq\bar{p}^* n'}{G; \Gamma \vdash_{n} T_1 \quad G; \Gamma \vdash_{n} T_2}{G; \Gamma \vdash_{n} T_1 \wedge \bar{p} \mathrel{\top} T_2} \quad \text{ANDBAR1}
\]
Soundness and completeness w.r.t. Pinto-Uustala 2009.

To simulate sequents $\Gamma \vdash_G \Delta$, need:

$$
p T' @ n' \in \Gamma \\
G; \Gamma, \bar{p} T @ n \vdash_{\bar{n}'} \bar{p} T' \\
\frac{G; \Gamma \vdash_{\bar{n}} T}{G; \Gamma \vdash p T}
$$

$$
\bar{p} T' @ n' \in \Gamma \\
G; \Gamma, \bar{p} T @ n \vdash_{\bar{n}'} \bar{p} T' \\
\frac{G; \Gamma \vdash_{\bar{n}} T}{G; \Gamma \vdash p T}
$$

Relate DIL without cut but with axCut rules

Cut elimination (with axCut rules)
Towards a Dualized Type Theory

Term syntax

\[ i \in \{1, 2\} \]
\[ t ::= x \mid (t, t') \mid in_i t \mid \lambda x.t \mid \langle t, t' \rangle \mid \nu x.t \cdot t' \]

Type assignment rules based on DIL
Reduction rules based on cut elimination.

\[ \nu x.(t_1, t_2) \cdot in_i t \leadsto \nu x.t_i \cdot t \]
\[ \nu x.\langle \lambda y.t \rangle \cdot \langle t_1, t_2 \rangle \leadsto \nu x.t_1 \cdot \nu y.t \cdot t_2 \]
\[ \nu x.\nu y.t_1 \cdot t_2 \cdot t \leadsto \nu x.[t/y](t_1 \cdot t_2) \]

Also terminating recursion, recursive types \( \mu^p X. T \)
Desired: normalization, type preservation, \( p \)-canonicity without \( \rightarrow \bar{p} \)
Benefits of constructive duality

Control with canonicity

- Can implement control constructs like delimited continuations

New insights into coinduction

- Negative well-founded data \(\Rightarrow\) observations
  - A list of \(A\)'s has positive type \(\mu^+ X.\langle+\rangle \land_- (A \land_+ X)\)
  - A colist has positive type \(\mu^- X.\langle+\rangle \land_- (A \land_+ X)\), because
  - An observation of a colist has negative type \(\mu^- X.\langle+\rangle \land_- (A \land_+ X)\)

- Terminating recursion at polarity \(p\) with \(\mu^p\)

- Define coinductive data by negative recursion on observations

First-class patterns

- Negative data = observations = pattern match
- Data can support different sets of observations (views)
- Support \texttt{cons/snoc} with pattern matching!
Conclusion

- Goal: import constructive duality from logic to programming
- Dualized Intuitionistic Logic (DIL)
  - Dualized syntax $A | \langle p \rangle | T \land p T' | T \rightarrow p T'$
  - Proof rules for $G; \Gamma \vdash_p^T T$
- Next: metatheory, type theory
- *Programming with negative data*
Conclusion

- Goal: import constructive duality from logic to programming
- Dualized Intuitionistic Logic (DIL)
  - Dualized syntax $A \vdash \langle p \rangle \mid T \wedge_p T' \mid T \rightarrow_p T'$
  - Proof rules for $G; \Gamma \vdash^n_p T$
- Next: metatheory, type theory
- *Programming with negative data*

Let’s open up the other half of the universe!