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 $\bot$
  - $T \wedge T'$ and $T \vee T'$
- Duality clear in sequent calculus:

$$\frac{}{\Gamma \vdash \top, \Delta} \qquad \frac{}{\Gamma, \bot \vdash \Delta}$$

$$\frac{\Gamma \vdash T, \Delta \quad \Gamma \vdash T', \Delta}{\Gamma \vdash T \wedge T', \Delta} \qquad \frac{\Gamma, T \vdash \Delta \quad \Gamma, T' \vdash \Delta}{\Gamma, T \vee T' \vdash \Delta}$$

$$\frac{\Gamma, T_i \vdash \Delta \quad i \in \{1,2\}}{\Gamma, T_1 \wedge T_2 \vdash \Delta} \qquad \frac{\Gamma \vdash T_i, \Delta \quad i \in \{1,2\}}{\Gamma \vdash T_1 \vee T_2 \vdash \Delta}$$

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

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

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

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

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

$$
\begin{array}{lcl}
\textit{polarities } p & ::= & + \mid - \\
\textit{formulas } T & ::= & A \mid \langle p \rangle \mid T \wedge_p T' \mid T \rightarrow_p T'
\end{array}
$$

  - $\langle + \rangle$ is $\top$
  - $\langle - \rangle$ is $\bot$
  - $A \wedge_+ B$ is $A \wedge B$
  - $A \wedge_- B$ is $A \vee B$
  - $A \rightarrow_+ B$ is $A \rightarrow B$
  - $A \rightarrow_- B$ is $B - A$

Semantics of formulas

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

$$
\begin{array}{lcl}
[\![A]\!]_w & \Leftrightarrow & A \in V(w) \\
[\![\langle + \rangle]\!]_w & \Leftrightarrow & \mathit{true} \\
[\![\langle - \rangle]\!]_w & \Leftrightarrow & \mathit{false} \\
[\![T \wedge_+ T']\!]_w & \Leftrightarrow & [\![T]\!]_w \wedge [\![T']\!]_w \\
[\![T \wedge_- T']\!]_w & \Leftrightarrow & [\![T]\!]_w \vee [\![T']\!]_w \\
[\![T \to_+ T']\!]_w & \Leftrightarrow & \forall w'. w \preccurlyeq w' \Rightarrow [\![T]\!]_{w'} \Rightarrow [\![T']\!]_{w'} \\
[\![T \to_- T']\!]_w & \Leftrightarrow & \exists w'. w \succcurlyeq w' \wedge \neg [\![T]\!]_{w'} \wedge [\![T']\!]_{w'}
\end{array}
$$

- Key fact: $\neg [\![T \to_- T']\!]_w \Leftrightarrow \forall w'. w \succcurlyeq w' \Rightarrow \neg [\![T]\!]_{w'} \Rightarrow \neg [\![T']\!]_{w'}$
- Monotonicity Theorem: $w \preccurlyeq w'$ and $[\![T]\!]_w$ implies $[\![T]\!]_{w'}$

Kripke models, classically

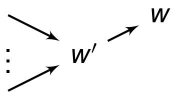
Define $\sim T \;:=\; T \to_{-} \langle + \rangle$

$[\![T \wedge_{-} \sim T]\!]_w = \mathit{true}$

"*Either T is true now, or there is an earlier world where it is false*"

$[\![\sim\sim T \to_{+} T]\!]_w = \mathit{true}$

"*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

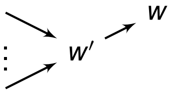
Define $\sim T \;:=\; T \rightarrow_{-} \langle + \rangle$

$[\![T \wedge_{-} \sim T]\!]_w = \mathit{true}$

"*Either T is true now, or there is an earlier world where it is false*"

$[\![\sim\sim T \rightarrow_{+} T]\!]_w = \mathit{true}$

"*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.

$$\begin{array}{lcl} \textit{world names (labels) } n & & \\ \textit{contexts } \Gamma & ::= & \cdot \mid \Gamma, p\, T@n \end{array}$$

- Treat Γ as a set
- Finite graphs G on world names
- Pinto, Uustalu's judgments: $\Gamma \vdash_G \Delta$ (unsigned Γ, Δ)
- 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,
  - $p\, [\![T]\!]_w$ holds

Proof rules

$$\frac{G \vdash n \preccurlyeq^{p*} n'}{G; \Gamma, p\, T@n \vdash^{p}_{n'} T}\ \text{AX} \qquad \frac{}{G; \Gamma \vdash^{p}_{n} \langle p \rangle}\ \text{UNIT}$$

$$\frac{\begin{array}{l} n' \notin |G| \\ (G, n \preccurlyeq^{p} n'); \Gamma, p\, T_1@n' \vdash^{p}_{n'} T_2 \end{array}}{G; \Gamma \vdash^{p}_{n} T_1 \rightarrow_p T_2}\ \text{IMP} \qquad \frac{\begin{array}{l} G \vdash n \preccurlyeq^{\bar{p}*} n' \\ G; \Gamma \vdash^{\bar{p}}_{n'} T_1 \quad G; \Gamma \vdash^{p}_{n'} T_2 \end{array}}{G; \Gamma \vdash^{p}_{n} T_1 \rightarrow_{\bar{p}} T_2}\ \text{IMPBAR}$$

$$\frac{G; \Gamma \vdash^{p}_{n} T_1 \quad G; \Gamma \vdash^{p}_{n} T_2}{G; \Gamma \vdash^{p}_{n} T_1 \wedge_p T_2}\ \text{AND} \qquad \frac{G; \Gamma \vdash^{p}_{n} T_1}{G; \Gamma \vdash^{p}_{n} T_1 \wedge_{\bar{p}} T_2}\ \text{ANDBAR1}$$

$$\frac{G; \Gamma \vdash^{p}_{n} T_2}{G; \Gamma \vdash^{p}_{n} T_1 \wedge_{\bar{p}} T_2}\ \text{ANDBAR2}$$

$$\frac{G; \Gamma, \bar{p}\, T@n \vdash^{p}_{n'} T' \quad G; \Gamma, \bar{p}\, T@n \vdash^{\bar{p}}_{n'} T'}{G; \Gamma \vdash^{p}_{n} T}\ \text{CUT}$$

Desired Metatheory

- Soundness and completeness w.r.t. Pinto-Uustala 2009.
  - To simulate sequents $\Gamma \vdash_G \Delta$, need:

$$\frac{p\, T'@n' \in \Gamma \qquad G; \Gamma, \bar{p}\, T@n \vdash^{\bar{p}}_{n'} T'}{G; \Gamma \vdash^{p}_{n} T}\ \textsc{axCut}$$

$$\frac{\bar{p}\, T'@n' \in \Gamma \qquad G; \Gamma, \bar{p}\, T@n \vdash^{p}_{n'} T'}{G; \Gamma \vdash^{p}_{n} T}\ \textsc{axCutBar}$$

  - Relate DIL without cut but with axCut rules
- Cut elimination (with axCut rules)

Towards a Dualized Type Theory

- Term syntax

$$\begin{array}{lcl} i & \in & \{1,2\} \\ t & ::= & x \mid (t,t') \mid in_i\ t \mid \lambda x.t \mid \langle t,t' \rangle \mid \nu x.t \bullet t' \end{array}$$

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

$$\begin{array}{lcl} \nu x.(t_1,t_2) \bullet in_i\ t & \rightsquigarrow & \nu x.t_i \bullet t \\ \nu x.(\lambda y.t) \bullet \langle t_1,t_2 \rangle & \rightsquigarrow & \nu x.t_1 \bullet \nu y.t \bullet t_2 \\ \nu x.(\nu y.t_1 \bullet t_2) \bullet t \rangle & \rightsquigarrow & \nu x.[t/y](t_1 \bullet t_2) \\ & \ldots & \end{array}$$

- 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 => *observations*
  - A list of *A*'s has positive type $\mu^{+} X.\langle + \rangle \wedge_{-} (A \wedge_{+} X)$
  - A colist has positive type $\mu^{-} X.\langle + \rangle \wedge_{-} (A \wedge_{+} X)$, because
  - An observation of a colist has negative type $\mu^{-} X.\langle + \rangle \wedge_{-} (A \wedge_{+} X)$
- Terminating recursion at polarity *p* with μ^{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 `cons`/`snoc` with pattern matching!

Conclusion

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

Conclusion

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

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