Duality and polarity in type theory

Noam Zeilberger

Carnegie Mellon University

23 May 2008

Programs $\longleftrightarrow$ Proofs

Programs	$\longleftrightarrow$	Proofs
typed lambda-calculus		natural deduction

Programs	$\longleftrightarrow$	Proofs
typed lambda-calculus	♡	natural deduction

The End

Programs	$\longleftrightarrow$	Proofs
typed lambda-calculus	♡	natural deduction

The End

Some unresolved plot threads?. . .

“Internal”: bureaucracy related to $\vee$-elim, etc.

“External”: missing features of real-world languages

- Effects and evaluation order (values, eval ctxs, etc.)
- *Operationally-sensitive typing phenomena*

The Visible Plumbing Problem

As types become more precise, detailed properties of the operational semantics become visible in the type system.

Polymorphism: value restriction in ML

$\cap$-types: value restriction
$(A \to B) \cap (A \to C) \leq A \to (B \cap C)$ unsafe!

$\cup$-types: “tridirectional typechecking”
$(A \to C) \cap (B \to C) \leq (A \cup B) \to C$ unsafe in CBN!

The Visible Plumbing Problem

As types become more precise, detailed properties of the operational semantics become visible in the type system.

Polymorphism: value restriction in ML

$\cap$-types: value restriction
$(A \to B) \cap (A \to C) \leq A \to (B \cap C)$ unsafe!

$\cup$-types: "tridirectional typechecking"
$(A \to C) \cap (B \to C) \leq (A \cup B) \to C$ unsafe in CBN!

Can we plug these leaks?

Outline of a solution

1. Distinction between *positive* and *negative* types (polarity)

Outline of a solution

1. Distinction between *positive* and *negative* types (polarity)
2. Define a language (logic) by first defining *patterns*
   - Positive types defined by *constructor* patterns con
   - Negative types defined by *destructor* patterns des

Outline of a solution

1. Distinction between *positive* and *negative* types (polarity)
2. Define a language (logic) by first defining *patterns*
   - Positive types defined by *constructor* patterns con
   - Negative types defined by *destructor* patterns des
3. Define terms (proofs) by reference to patterns

	Val	Kon
positive (“strict”)	$\mathrm{con} \times \mathrm{Sub}$	$\mathrm{con} \to \mathrm{Stm}$
negative (“lazy”)	$\mathrm{des} \to \mathrm{Stm}$	$\mathrm{des} \times \mathrm{Sub}$

Outline of a solution

1. Distinction between *positive* and *negative* types (polarity)
2. Define a language (logic) by first defining *patterns*
   - Positive types defined by *constructor* patterns con
   - Negative types defined by *destructor* patterns des
3. Define terms (proofs) by reference to patterns

	Val	Kon
positive ("strict")	$\mathrm{con} \times \mathrm{Sub}$	$\mathrm{con} \to \mathrm{Stm}$
negative ("lazy")	$\mathrm{des} \to \mathrm{Stm}$	$\mathrm{des} \times \mathrm{Sub}$

4. *Derive* operationally-sensitive phenomena

Outline of a solution

1. Distinction between *positive* and *negative* types (polarity)
2. Define a language (logic) by first defining *patterns*
   - Positive types defined by *constructor* patterns con
   - Negative types defined by *destructor* patterns des
3. Define terms (proofs) by reference to patterns

	Val	Kon
positive ("strict")	$\mathrm{con} \times \mathrm{Sub}$	$\mathrm{con} \to \mathrm{Stm}$
negative ("lazy")	$\mathrm{des} \to \mathrm{Stm}$	$\mathrm{des} \times \mathrm{Sub}$

4. *Derive* operationally-sensitive phenomena

Positive types

An introduction rule gives, so to say, a definition of the constant in question. (Gentzen)

Positive types

An introduction rule gives, so to say, a definition of the constant in question. (Gentzen)

Define connectives via *linear* right-rules + axioms:

$$\frac{}{\cdot \Vdash 1} \qquad \frac{\Delta_1 \Vdash A \quad \Delta_2 \Vdash B}{\Delta_1, \Delta_2 \Vdash A \otimes B}$$

(no rule for 0)

$$\frac{\Delta \Vdash A}{\Delta \Vdash A \oplus B} \qquad \frac{\Delta \Vdash B}{\Delta \Vdash A \oplus B}$$

$$\frac{}{\neg A \Vdash \neg A}$$

Positive types

An introduction rule gives, so to say, a definition of the constant in question. (Gentzen)

Define connectives via *linear* right-rules + axioms:

$$\frac{}{\cdot \Vdash 1} \qquad \frac{\Delta_1 \Vdash A \quad \Delta_2 \Vdash B}{\Delta_1, \Delta_2 \Vdash A \otimes B}$$

(no rule for 0)

$$\frac{\Delta \Vdash A}{\Delta \Vdash A \oplus B} \qquad \frac{\Delta \Vdash B}{\Delta \Vdash A \oplus B}$$

$$\frac{}{\neg A \Vdash \neg A}$$

Positive types

An introduction rule gives, so to say, a definition of the constant in question. (Gentzen)

Define connectives via *linear* right-rules + axioms:

$$\frac{}{\cdot \Vdash () : 1} \qquad \frac{\Delta_1 \Vdash p_1 : A \quad \Delta_2 \Vdash p_2 : B}{\Delta_1, \Delta_2 \Vdash (p_1, p_2) : A \otimes B}$$

(no rule for 0)

$$\frac{\Delta \Vdash p : A}{\Delta \Vdash \mathsf{inl}\, p : A \oplus B} \qquad \frac{\Delta \Vdash p : B}{\Delta \Vdash \mathsf{inr}\, p : A \oplus B}$$

$$\frac{}{\kappa \Leftarrow A \Vdash \kappa : \neg A}$$

Usual pattern restrictions emerge!

CBV continuation calculus

Typing contexts

$\Delta ::= \cdot \mid \Delta, \kappa \Leftarrow A$

$\Gamma ::= \cdot \mid \Gamma, \Delta$

Judgments

$\Gamma \vdash V : A$	value
$\Gamma \vdash K \Leftarrow A$	CBV continuation
$\Gamma \vdash S : \#$	computation
$\Gamma \vdash \sigma : \Delta$	substitution

(Define these generically, using judgment $\Delta \Vdash p : A$)

(Positive) values

A value is just a pattern under a substitution

$$\frac{\Delta \Vdash p : A \quad \Gamma \vdash \sigma : \Delta}{\Gamma \vdash p[\sigma] : A}$$

(Positive) values

A value is just a pattern under a substitution

$$\frac{\Delta \Vdash p : A \quad \Gamma \vdash \sigma : \Delta}{\Gamma \vdash p[\sigma] : A}$$

(Replace “is just” by “uniquely factors as”?)

(Positive) values

A value is just a pattern under a substitution

$$\frac{\Delta \Vdash p : A \quad \Gamma \vdash \sigma : \Delta}{\Gamma \vdash p[\sigma] : A}$$

(Replace “is just” by “uniquely factors as”?)

E.g., derived rule for $A = \neg B \oplus (\neg C_1 \otimes \neg C_2)$:

$$\frac{\Gamma \vdash K_1 \Leftarrow C_1 \quad \Gamma \vdash K_2 \Leftarrow C_2}{\Gamma \vdash \mathsf{inr}(\kappa_1, \kappa_2)[K_1/\kappa_1, K_2/\kappa_2] : A} \; (\kappa_1 \Leftarrow C_1, \kappa_2 \Leftarrow C_2 \Vdash \mathsf{inr}(\kappa_1, \kappa_2) : A)$$

(Positive) values

A value is just a pattern under a substitution

$$\frac{\Delta \Vdash p : A \quad \Gamma \vdash \sigma : \Delta}{\Gamma \vdash p[\sigma] : A}$$

(Replace “is just” by “uniquely factors as”?)

E.g., derived rule for $A = \neg B \oplus (\neg C_1 \otimes \neg C_2)$:

$$\frac{\Gamma \vdash K_1 \Leftarrow C_1 \quad \Gamma \vdash K_2 \Leftarrow C_2}{\Gamma \vdash \mathsf{inr}(K_1, K_2) : A} \; (\kappa_1 \Leftarrow C_1, \kappa_2 \Leftarrow C_2 \Vdash \mathsf{inr}(\kappa_1, \kappa_2) : A)$$

(Positive) continuations

A continuation is a *map* from patterns to statements

$$\frac{\forall(\Delta \Vdash p : A) : \quad \Gamma, \Delta \vdash \phi(p) : \#}{\Gamma \vdash (\phi) \Leftarrow A}$$

(Positive) continuations

A continuation is a *map* from patterns to statements

$$\frac{\forall(\Delta \Vdash p : A) : \quad \Gamma, \Delta \vdash \phi(p) : \#}{\Gamma \vdash (\phi) \Leftarrow A}$$

(Is this HOAS or AHOS?)

(Positive) continuations

A continuation is a *map* from patterns to statements

$$\frac{\forall(\Delta \Vdash p : A) : \quad \Gamma, \Delta \vdash \phi(p) : \#}{\Gamma \vdash (\phi) \div A}$$

(Is this HOAS or AHOS?)

E.g., derived rule for $A = \neg B \oplus (\neg C_1 \otimes \neg C_2)$:

$$\frac{\Gamma, \kappa \div B \vdash S_1 : \# \quad \Gamma, \kappa_1 \div C_1, \kappa_2 \div C_2 \vdash S_2 : \#}{\Gamma \vdash \{\mathsf{inl}(\kappa) \mapsto S_1 \mid \mathsf{inr}(\kappa_1, \kappa_2) \mapsto S_2\} \div A}$$

Substitutions and statements

$$\frac{}{\Gamma \vdash (\cdot) : (\cdot)} \qquad \frac{\Gamma \vdash \sigma : \Delta \quad \Gamma \vdash K \Leftarrow A}{\Gamma \vdash (\sigma, K/\kappa) : (\Delta, \kappa \Leftarrow A)}$$

$$\frac{\kappa \Leftarrow A \in \Gamma \quad \Gamma \vdash V : A}{\Gamma \vdash \kappa\, V : \#}$$

Substitutions and statements

$$\frac{}{\Gamma \vdash (\cdot) : (\cdot)} \qquad \frac{\Gamma \vdash \sigma : \Delta \quad \Gamma \vdash K \Leftarrow A}{\Gamma \vdash (\sigma, K/\kappa) : (\Delta, \kappa \Leftarrow A)}$$

$$\frac{\kappa \Leftarrow A \in \Gamma \quad \Gamma \vdash V : A}{\Gamma \vdash \kappa\, V : \#}$$

(Note: could make these rules linear.)

Substitutions and statements

$$\frac{}{\Gamma \vdash (\cdot) : (\cdot)} \qquad \frac{\Gamma \vdash \sigma : \Delta \quad \Gamma \vdash K \Leftarrow A}{\Gamma \vdash (\sigma, K/\kappa) : (\Delta, \kappa \Leftarrow A)}$$

$$\frac{\kappa \Leftarrow A \in \Gamma \quad \Gamma \vdash V : A}{\Gamma \vdash \kappa\, V : \#}$$

(Note: could make these rules linear.)

Add *cut principles* for computation:

- If $\Gamma \vdash K \Leftarrow A$ and $\Gamma \vdash V : A$ then $\Gamma \vdash K\, V : \#$
- If $\Gamma, \Delta \vdash t : J$ and $\Gamma \vdash \sigma : \Delta$ then $\Gamma \vdash t[\sigma] : J$

Operational semantics:

$$(\phi)\,(p[\sigma]) \rightsquigarrow \phi(p)[\sigma]$$

A purely interactive approach to logic

« Interactive » could suggest yet-one-more-game-semantics : but the material presented here is neither syntax nor semantics, moreover the word *purely* suggests a distance with the mere idea of game : there is no rule —or no referee, if you prefer— like in real life. And *logic*, without « s », is for what should be the most natural thing in nature —something too often presented as the most artificial one.

The monograph ends with a dictionary, discussing these issues : sort of final introduction, since one can only introduce to known material. For instance if you go to DIALECTICS you will understand the word

Ludics

which is the real alternative title, the very name of the new area.

The novelty of ludics is conveyed by our title

Locus Solum

after the book by Raymond Roussel, *Locus Solus*, i.e., « solitary place ». *Locus Solum* means something like

Only the location matters

for the results presented here establish the pregnancy of location, the *locus*, in logic. As you will see, the irruption of the *locus* by no way weakens or dilutes logical principles : they just become different, more harmonious, and stronger. Moreover the logic-we-used-to-know-and-love is still present, but it now gets a specific name, *spiritual logic* : ludics created spiritual logic in the same way Brouwer created classical logic and Luther catholicism.

A purely interactive approach to logic

« Interactive » could suggest yet-one-more-game-semantics : but the material presented here is neither syntax nor semantics, moreover the word *purely* suggests a distance with the mere idea of game : there is no rule —or no referee, if you prefer— like in real life. And *logic*, without « s », is for what should be the most natural thing in nature —something too often presented as the most artificial one.

The monograph ends with a dictionary, discussing these issues : sort of final introduction, since one can only introduce to known material. For instance if you go to DIALECTICS you will understand the word

Ludics

which is the real alternative title, the very name of the new area.

The novelty of ludics is conveyed by our title

Locus Solum

after the book by Raymond Roussel, *Locus Solus*, i.e., « solitary place ». *Locus Solum* means something like

Only the location matters

for the results presented here establish the pregnancy of location, the *locus*, in logic. As you will see, the irruption of the *locus* by no way weakens or dilutes logical principles : they just become different, more harmonious, and stronger. Moreover the logic-we-used-to-know-and-love is still present, but it now gets a specific name, ***spiritual logic*** : ludics created spiritual logic in the same way Brouwer created classical logic and Luther catholicism.

How do you always stay so positive?

Recall Gentzen's remark: *An introduction rule gives, so to say, a definition of the constant in question.*

How do you always stay so positive?

Recall Gentzen's remark: *An introduction rule gives, so to say, a definition of the constant in question.*

. . . [But that] has no more force than the converse suggestion, that they are fixed by the elimination rules. (Dummett)

How do you always stay so positive?

Recall Gentzen's remark: *An introduction rule gives, so to say, a definition of the constant in question.*

. . . [But that] has no more force than the converse suggestion, that they are fixed by the elimination rules. (Dummett)

E.g.,:

- Function space $A \to B$ defined by application
- Lazy pairs $A \& B$ defined by projections

How do you always stay so positive?

Recall Gentzen's remark: *An introduction rule gives, so to say, a definition of the constant in question.*

. . . [But that] has no more force than the converse suggestion, that they are fixed by the elimination rules. (Dummett)

E.g.,:

- Function space $A \to B$ defined by application
- Lazy pairs $A \& B$ defined by projections

But first we need some bureaucracy. . .

Polarized types

Distinguish positive and negative types

$$A^+ ::= A^+ \oplus A^+ \mid A^+ \otimes A^+ \mid \overset{v}{\neg} A^+ \mid \downarrow A^-$$

$$A^- ::= A^- \& A^- \mid A^+ \to A^- \mid \overset{n}{\neg} A^- \mid \uparrow A^+$$

($\downarrow$ and $\uparrow$ are "shifts")

Typing contexts

$$\Delta ::= \cdot \mid \Delta, \kappa \div A^+ \mid \Delta, u : A^-$$

Notation: can omit polarity of well-polarized type A

Negative types

Defined via linear left-rules (destructor patterns):

$$\frac{\Delta_1 \Vdash A \quad \Delta_2; B \Vdash \cdot}{\Delta_1, \Delta_2; A \to B \Vdash \cdot}$$

$$\frac{\Delta; A \Vdash \cdot}{\Delta; A \& B \Vdash \cdot} \qquad \frac{\Delta; B \Vdash \cdot}{\Delta; A \& B \Vdash \cdot}$$

$$\frac{}{A; \overset{n}{\neg} A \Vdash \cdot}$$

Negative types

Defined via linear left-rules (destructor patterns):

$$\frac{\Delta_1 \Vdash A \quad \Delta_2; B \Vdash \cdot}{\Delta_1, \Delta_2; A \to B \Vdash \cdot}$$

$$\frac{\Delta; A \Vdash \cdot}{\Delta; A \& B \Vdash \cdot} \qquad \frac{\Delta; B \Vdash \cdot}{\Delta; A \& B \Vdash \cdot}$$

$$\frac{}{A; \overset{n}{\neg} A \Vdash \cdot}$$

Negative types

Defined via linear left-rules (destructor patterns):

$$\frac{\Delta_1 \Vdash p : A \quad \Delta_2 \Vdash d \Leftarrow B}{\Delta_1, \Delta_2 \Vdash \mathsf{app}(p); d \Leftarrow A \to B}$$

$$\frac{\Delta \Vdash d \Leftarrow A}{\Delta \Vdash \mathsf{fst}; d \Leftarrow A \& B} \qquad \frac{\Delta \Vdash d \Leftarrow B}{\Delta \Vdash \mathsf{snd}; d \Leftarrow A \& B}$$

$$\frac{}{u : A \Vdash u \Leftarrow \overset{n}{\neg} A}$$

Shifts

$$\frac{}{u : A \Vdash u : \downarrow A} \qquad \frac{}{\kappa \Leftarrow A \Vdash \kappa \Leftarrow \uparrow A}$$

Shifts

$$\frac{}{u : A \Vdash u : \downarrow A} \qquad \frac{}{\kappa \Div A \Vdash \kappa \Div \uparrow A}$$

What do the shifts *mean?*

- Think $A \xrightarrow{v} B = A \to \uparrow B$, $A \xrightarrow{n} B = \downarrow A \to B$
- But bear with me. . .

Negative values

A map from destructor patterns to statements

$$\frac{\forall(\Delta \Vdash d \doteqdot A) : \quad \Gamma, \Delta \vdash \psi(d) : \#}{\Gamma \vdash (\psi) : A}$$

Negative values

A map from destructor patterns to statements

$$\frac{\forall(\Delta \Vdash d \div A): \quad \Gamma, \Delta \vdash \psi(d) : \#}{\Gamma \vdash (\psi) : A}$$

E.g., derived rule for $A = \downarrow B \to (\uparrow C_1 \& \uparrow C_2)$:

$$\frac{\Gamma, u : B, \kappa \div C_1 \vdash S_1 : \# \quad \Gamma, u : B, \kappa \div C_2 \vdash S_2 : \#}{\Gamma \vdash \{\mathsf{app}(u); \mathsf{fst}; \kappa \mapsto S_1 \mid \mathsf{app}(u); \mathsf{snd}; \kappa \mapsto S_2\} : A}$$

Negative continuations

A destructor pattern under a substitution

Negative continuations

A destructor pattern under a substitution

(You can figure out the rest.)

Shifting back

A negative value $V : \uparrow A$ is a *computation*

$$\frac{\Gamma, \kappa \div A \vdash S : \#}{\Gamma \vdash \{\kappa \mapsto S\} : \uparrow A}$$

Notation $E : \uparrow A$

(A positive value $V : \downarrow\uparrow A$ is a *suspended* computation)

Shifting back

A negative value $V : \uparrow A$ is a *computation*

$$\frac{\Gamma, \kappa \Leftarrow A \vdash S : \#}{\Gamma \vdash \{\kappa \mapsto S\} : \uparrow A}$$

Notation $E : \uparrow A$

(A positive value $V : \downarrow\uparrow A$ is a *suspended* computation)

Can derive computation-typing rules, e.g.:

$$\frac{\Gamma \vdash E_1 : \uparrow A \quad \Gamma \vdash E_2 : \uparrow B}{\Gamma \vdash (E_1, E_2) : \uparrow(A \otimes B)}$$

where (E_1, E_2) is the left-to-right CBV CPS transform

Conclusions

What do we have? (See paper in APAL)

- A pattern-centric view of syntax and semantics
- A language with both positive and negative types
- An isomorphism with proofs in *focused sequent calculus*

What else do we have?

- A type system with $\cap$, $\cup$ and effects (forthcoming)
- An explanation of HOAS (Licata, Z & Harper in LICS08)

What can we hope for?

- Π and Σ? Multiple modalities? Topological semantics?

Programs $\longleftrightarrow$ Proofs

The End