Adjoint Logic and Its Concurrent Semantics

Frank Pfenning
ABCD Meeting, Edinburgh, December 18-19, 2017
Joint work with Klaas Pruiksma and William Chargin
Outline

• Proofs as programs
  • Linear sequent proofs as concurrent programs
    • Take 1: standard sequent calculus = synchronous communication
    • Take 2: unary $\otimes R$ and $\lnot oR$
    • Take 3: positive axiomatic form = asynchronous communication

• Adjoint logic
  • Structural properties and modes of truth
  • Declaration of Independence

• Concurrent operational semantics
  • Contraction, weakening, multicut, and multi-identity
  • Garbage collection
Proofs as Programs

• Codesign of language and its reasoning principles
• Three levels of correspondence
  • Propositions as types
  • Proofs as programs
  • Proof reduction as computation
• Style of proof system is critical to characterize computation
  • Axiomatic style ⇔ combinators and combinatory reduction [Curry’35]
  • Natural deduction ⇔ λ-calculus and substitution [Howard’69]
  • Sequent calculus ⇔ explicit substitutions [Herbelin’94]
• All intuitionistic and structural (admitting weakening & contraction)
Linear Proofs as Session-Typed Programs

• Three levels of correspondence
  • Linear propositions as session types
  • Sequent proofs as concurrent programs
  • Cut reduction as communication

• Intuitionistic variant: provider/client model [Caires & Pf’10]
  • No need to dualize types
  • Dependent types [Toninho et al.’11]
  • Monadic integration with functional language (SILL) [Toninho’15] [Griffith’15]
  • Integration in imperative language (CC0) [Willsey et al.16]
  • Polymorphism and logical relations [Perez et al.’13]
  • Exploiting categorical view (this talk)

• Classical variant: symmetric communication
  • [Wadler’12] [Caires et al.’16]
Outline

• Proofs as programs
• Linear sequent proofs as concurrent programs
  • Take 1: standard sequent calculus = synchronous communication
  • Take 2: unary $\otimes R$ and $\neg oR$
  • Take 3: positive axiomatic form = asynchronous communication
• Adjoint logic
  • Structural properties and modes of truth
  • Declaration of Independence
• Concurrent operational semantics
  • Contraction, weakening, multicut, and multi-identity
  • Garbage collection
Cut as Parallel Composition

\[ \Delta \vdash c : A \quad \Delta', c : A \vdash e : C \]
\[ \Delta, \Delta' \vdash e : C \]

process configuration always forms a tree
Cut as Process Spawn

\[ \Delta \vdash c : A \quad \Delta', c : A \vdash e : C \]

\[ \Delta, \Delta' \vdash e : C \]

cut
Identity as Identification

\[ d : A \vdash c : A \quad \text{id} \]

transition of configuration models
cut of any proof with identity
\[
\Delta \vdash x : A \quad \Delta' \vdash c : B \\
\frac{}{\Delta, \Delta' \vdash c : A \otimes B}
\]

\[
\otimes R
\]

Tensor, Original
Tensor, Simplified

\[ \Delta \vdash c : B \]
\[ \Delta, d : A \vdash c : A \otimes B \]
\[ \otimes R^* \]

cut reduction predisposes to synchronous semantics
Interderivable using Identity and Cut

\[
\begin{align*}
\frac{A \vdash A}{\Delta, A \vdash A \otimes B} \quad \otimes R
\end{align*}
\]

\[
\begin{align*}
\frac{\Delta \vdash A \quad \Delta' \vdash B}{\Delta, \Delta' \vdash A \otimes B} \quad \otimes R^* \\
\frac{\Delta' \vdash B \quad \Delta', A \vdash A \otimes B}{\Delta, \Delta' \vdash A \otimes B} \quad \text{cut}
\end{align*}
\]
Tensor as a Message

use axiomatic form for positive providers or negative clients ("senders")

\[
d : A, c' : B \vdash c : A \otimes B \otimes RA
\]

asynchronous message
sending \(d : A\) along \(c : A \otimes B\)
with continuation \(c' : B\)
Interderivable with 2 cuts, 2 ids

\[
\frac{A \vdash A \text{ id} \quad B \vdash B \text{ id}}{A, B \vdash A \otimes B \quad \otimes R}
\]

because \( \otimes RA \) has no continuation
an asynchronous semantics seems forced!

\[
\frac{\Delta' \vdash B \quad A, B \vdash A \otimes B \quad \otimes RA \text{ cut}}{\Delta, \Delta' \vdash A \otimes B}
\]

\[
\frac{\Delta \vdash A \quad \Delta', A \vdash A \otimes B \quad \text{cut}}{\Delta, \Delta' \vdash A \otimes B}
\]
Positive Axiomatic Formulation

• Forces (?) asynchronous semantics
• Restore synchronous communication via mode-neutral shifts [Pf & Griffith’15]
• Can write all “sending” rules as axioms
  • Right rules for positive connectives
  • Left rules for negative connectives
• Can hide this from surface syntax, if desired
Outline

• Proofs as programs
• Linear sequent proofs as concurrent programs
  • Take 1: standard sequent calculus = synchronous communication
  • Take 2: unary $\otimes$R and $-oR$
  • Take 3: positive axiomatic form = asynchronous communication
• Adjoint logic
  • Structural properties and modes of truth
  • Declaration of Independence
• Concurrent operational semantics
  • Contraction, weakening, multicut, and multi-identity
  • Garbage collection
Structural Rules

- Weakening: do not need to use antecedents (affine logic)
- Contraction: may reuse antecedents (strict logic)
- Weakening + contraction = structural logic

\[
\frac{\Delta \vdash e : C}{\Delta, c : A \vdash e : C'} \quad \text{W} \quad \frac{\Delta, c_1 : A, c_2 : A \vdash e : C}{\Delta, c : A \vdash e : C'}
\]

- Compromises the usual cut reduction and cut elimination!
Adjoint Logic [Reed’09]

• Would like to have our cake and eat it, too!
  • Allow weakening, contraction where desirable or necessary

• Challenges
  • How do we make system coherent: linear remains linear, etc.
  • Combination should be conservative over its parts
  • Logically: cut elimination, identity elimination
  • Operationally: session fidelity, global progress (even with recursion)
  • Uniform syntax and semantics
Modes of Truth

• Modes of truth k, m, n
• Every proposition $A_m$ has an intrinsic mode of truth $m$
• Every mode $m$ possesses structural properties $\sigma(m) \subseteq \{W, C\}$
  • It is possible to add exchange as an option, starting from ordered logic
• Modes are related by preorder $\leq$, where $m \leq k$ implies $\sigma(m) \subseteq \sigma(k)$
Modes of Truth, continued

• Syntax of propositions (and proofs) is uniform across all modes

\[ A_m, B_m ::= P_m \mid A_m \rightarrow B_m \mid \&\{ \ell : A^\ell_m \}_\ell \in L \mid \uparrow^m_k A_k \]
\[ \mid A_m \otimes B_m \mid \mathbf{1} \mid \oplus\{ \ell : A^\ell_m \}_\ell \in L \mid \downarrow^n_m A_n \]

• Shift \( \uparrow^m_k A_k \) references proposition \( A_k \) in mode \( m \), for \( m \geq k \)
• Shift \( \downarrow^n_m A_n \) references proposition \( A_n \) in mode \( m \), for \( n \geq m \)
Example: Intuitionistic Linear Logic

- Unrestricted mode $U$ with $\sigma(U) = \{W,C\}$
- Linear mode $L$ with $\sigma(L) = \{\}$
- $L < U$
- Mode $U$ populated only by shifts

\[
\begin{align*}
A_U & ::= \uparrow^U_L A_L \\
A_L & ::= A_L \rightsquigarrow B_L \mid \cdots \mid \downarrow^U_L A_U \\
!A_L & \triangleq \downarrow^U_L \uparrow^U_L A_L
\end{align*}
\]
Example: LNL [Benton’94]

• Unrestricted mode U with $\sigma(U) = \{W,C\}$, with all connectives
• Linear mode L with $\sigma(L) = \{\}$
• $L < U$

$m ::= L \mid U$ with $L < U$

$A_m, B_m ::= P_m \mid A_m \to B_m \mid \&\{\ell:A_\ell\}_{\ell \in L} \mid U^L_A L \\
\| A_m \otimes B_m \mid 1 \mid \oplus\{\ell:A_\ell\}_{\ell \in L} \mid U^L_A U$

$A_U \to B_U \triangleq A_U \to B_U$
Other Examples

• Intuitionistic S4 ~ staged computation
  • U < V, σ(U) = σ(V) = {W,C}

• Lax logic ~ monadic encapsulation
  • X < U, σ(X) = σ(U) = {W,C}

• Subexponential logic
  • Like adjoint logic, distinguished mode L
  • All other modes m contain only \( \uparrow_L^m A_L \)
  • \( !^m A_L \triangleq \downarrow_L^m \uparrow_L^m A_L \)
The Declaration of Independence

• Key to obtaining coherence, conservativity, cut elimination, uniformity
• \( \Psi ::= \varepsilon \mid \mathcal{A}_m \mid \Psi_1, \Psi_2 \) where ‘,’ is associative, commutative, w/unit \( \varepsilon \)
• \( \Psi \geq m \) if \( k \geq m \) for every \( \mathcal{A}_k \) in \( \Psi \)

**\[ \Psi \vdash \mathcal{A}_m \text{ is well-formed only if } \Psi \geq m \]**

• Truth of \( \mathcal{A}_m \) must be independent from all \( \mathcal{A}_k \) for \( k \neq m \)
Rules for Adjoint Logic

• Logical rules unchanged, stay at same, but arbitrary mode \( m \)
• Structural rules depend on mode properties

\[
\begin{align*}
W \in \sigma(m) & \quad \Delta \vdash e : C_r \\
\Delta, c : A_m & \vdash e : C_r
\end{align*}
\]

\[
\begin{align*}
C \in \sigma(m) & \quad \Delta, c_1 : A_m, c_2 : A_m \vdash e : C \\
\Delta, c : A_m & \vdash e : C_r
\end{align*}
\]
Rules for Shift

- Derived entirely from declaration of independence

\[
\frac{\Psi \vdash A_k}{\Psi \vdash \uparrow^m_k A_k} \quad \uparrow R
\]

\[
\frac{\Psi \geq n \quad \Psi \vdash A_n}{\Psi \vdash \downarrow^n_m A_n} \quad \downarrow R
\]

\[
\frac{k \geq r \quad \Psi, A_k \vdash C_r}{\Psi, \uparrow^m_k A_k \vdash C_r} \quad \uparrow L
\]

\[
\frac{\Psi, A_n \vdash C_r}{\Psi, \downarrow^n_m A_n \vdash C_r} \quad \downarrow L
\]
Judgmental Rules

• Cut and identity are the most interesting rules
• Standard argument for cut elimination does not work in the presence of explicit contraction
• Return to Gentzen (1935): Multicut
Multicut

• Some restrictions forced by declaration of independence

\[ \bar{c} = (c_1, \ldots, c_n) \]
\[ \Psi \geq m \geq r \quad \Psi \vdash \bar{c} : A_m \quad \Psi', c_1 : A_m, \ldots, c_n : A_m \vdash e : C_r \]
\[ \Psi, \Psi' \vdash e : C_r \] mcut*

• If \( W \in \sigma(m) \), then \( n = 0 \) is allowed
• If \( C \in \sigma(m) \), then \( n > 1 \) is allowed
• Provider is “aware of” all client channels \( \bar{c} \)
Multi-identity

• Provider might be communicating with multiple clients

\[
d : A_m \vdash \overline{c} : A_m \quad \text{id}^*
\]

• If \( W \in \sigma(m) \), then \( |\overline{c}| = 0 \) is allowed
• If \( C \in \sigma(m) \), then \( |\overline{c}| > 1 \) is allowed
Outline

• Proofs as programs
• Linear sequent proofs as concurrent programs
  • Take 1: standard sequent calculus = synchronous communication
  • Take 2: unary $\otimes R$ and $\neg oR$
  • Take 3: positive axiomatic form = asynchronous communication
• Adjoint logic
  • Structural properties and modes of truth
  • Declaration of Independence
• Concurrent operational semantics
  • Contraction, weakening, multicut, and multi-identity
  • Garbage collection
Process Interpretation

• Transitions do not care about modes at runtime
• However, some rules care about the number of clients a process has
• In this uniform semantics, computation at all structural properties is implemented by message passing
• Transitions of process configurations mimic cut reductions (not cut elimination)
• Shifts send and receive ’shift’ messages which synchronize [Pf & Griffith’15]
Multicut, Process Interpretation

one provider with multiple clients

\[
\Delta \vdash \overline{c} : A \\
\Delta', c_1:A, \ldots, c_n:A \vdash e : C
\]

\[
\Delta, \Delta' \vdash e : C
\]

mcut

\[
\overline{c} = c_1, \ldots, c_n
\]

provider should know about all client channels

without modes, to emphasize dynamic mode independence

12/19/17 ABCD Meeting, Edinburgh
Key Idea: Contraction = Identity + Multicut

\[
\frac{c : A \vdash \overline{c} : A}{\Delta, c_1 : A, c_2 : A \vdash e : C} \quad \text{id} \\
\frac{\Delta, c : A \vdash e : C}{\text{mcut}}
\]
Key Idea: Weakening = Identity + Multicut
Key Idea: Identity Propagation

- Helps to implement both drop (W) and copy (C)
- Distributed garbage collection “for free”
Key Ideas: Multicast and Copy-on-Receive

• Positive types (e.g., $A \otimes B$) do multicast, if there are multiple clients
• Negative types (e.g., $A \rightarrow{o} B$) copy-on-receive, if there are multiple clients
• Previous work on $!A$ only performed copy-on-receive
Ongoing and Future Work

• Exploit independence principle for non-uniform semantics
  • $S = \text{shared}, U = \text{unrestricted}, L = \text{linear}$ with $S \leq U, U \leq S, L < U, L < S$ and $\sigma(S) = \sigma(U) = \{C,W\}, \sigma(L) = \{\}$ [Balzer & Pf, 2017]
  • $H = \text{locally heap-allocated}, L = \text{linear}, L < H$ with $\sigma(H) = ?, \sigma(L) = \{\}$
  • Proof irrelevance, intensional equality [Pf’01]
  • Ghost messages for contracts and reasoning about distributed computation

• Is there a general theorem about non-uniform compatibility?
• Surface syntax? (Uniform is possible, up to a point)
• General implementation
Summary

- Adjoint logic combines logics coherently and conservatively
- Shifts compose to a monad (one order) and comonad (other order)
- Declaration of Independence enables key results
  - Cut elimination, identity elimination
  - Conservative extension over combined fragments
- Uniform message-passing semantics via multicut and identity
  - Contraction = identity + multicut for $n > 1$
  - Weakening = identity + multicut for $n = 0$
  - Distributed garbage collection = identity propagation