Higher-Order Focusing on Linearity and Effects Siva Somayyajula

6/11/2025

Focused and effect calculi share polarization of types

- Focused and effect calculi share polarization of types
- In call-by-push-value and friends: value and computation types

- Focused and effect calculi share polarization of types
- In call-by-push-value and friends: value and computation types
- Focusing has positive and negative formulae

- Focused and effect calculi share polarization of types
- In call-by-push-value and friends: value and computation types
- Focusing has positive and negative formulae
- This talk:

- Focused and effect calculi share polarization of types
- In call-by-push-value and friends: value and computation types
- Focusing has positive and negative formulae
- This talk:
 - Existing correspondence, e.g., between focused intuitionistic logic and callby-push-value (CBPV)

- Focused and effect calculi share polarization of types
- In call-by-push-value and friends: value and computation types
- Focusing has positive and negative formulae
- This talk:
 - Existing correspondence, e.g., between focused intuitionistic logic and callby-push-value (CBPV)
 - A higher-order focused analogue of the enriched effect calculus (EEC)

• Distinguish formulas (A) by **polarity**: **positive** (P, Q) and **negative** (N, O).

• Distinguish formulas (A) by **polarity**: **positive** (P, Q) and **negative** (N, O). • Intuitionistic sequent calculus: non-backtracking left rules (+) vs. right rules (-)

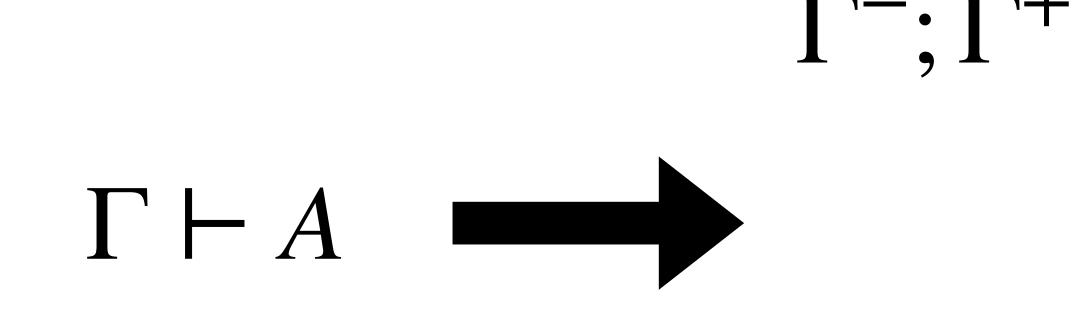
$\Gamma \vdash A$

• Distinguish formulas (A) by **polarity**: **positive** (P, Q) and **negative** (N, O). • Intuitionistic sequent calculus: non-backtracking left rules (+) vs. right rules (-)



• Distinguish formulas (A) by **polarity**: **positive** (P, Q) and **negative** (N, O). • Intuitionistic sequent calculus: non-backtracking left rules (+) vs. right rules (-)

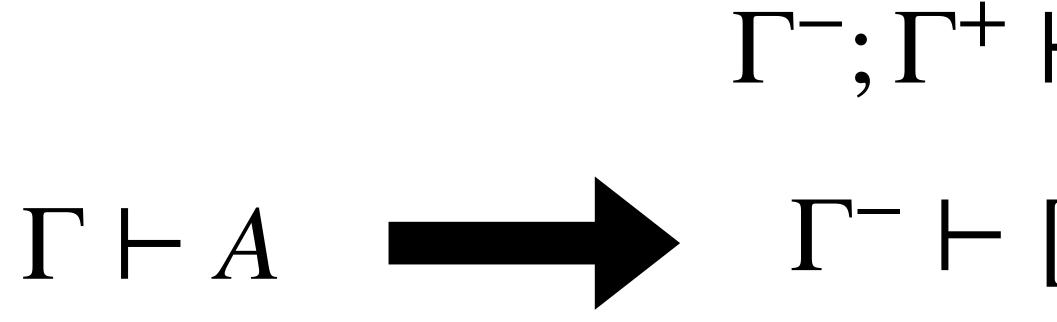
- Distinguish formulas (A) by **polarity**: **positive** (P, Q) and **negative** (N, O).
- Intuitionistic sequent calculus: non-backtracking left rules (+) vs. right rules (-)



 $\Gamma^-; \Gamma^+ \vdash A$ inversion (non-backtracking rules)

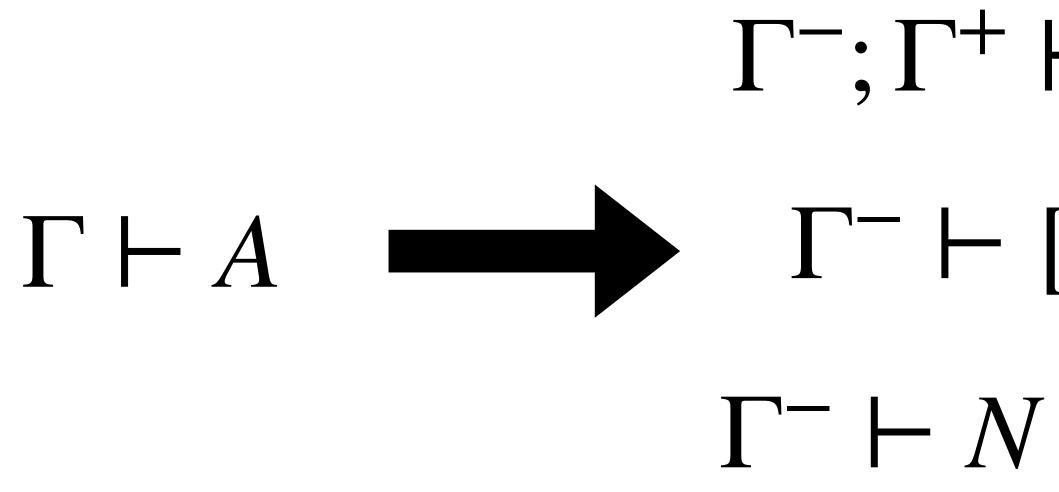
)

- Distinguish formulas (A) by polarity: positive (P, Q) and negative (N, O). • Intuitionistic sequent calculus: non-backtracking left rules (+) vs. right rules (-) Γ^{-} ; $\Gamma^{+} \vdash A$ inversion (non-backtracking rules) $\Gamma \vdash A$ $\Gamma \vdash [P]$ right focus (backtracking right rules)





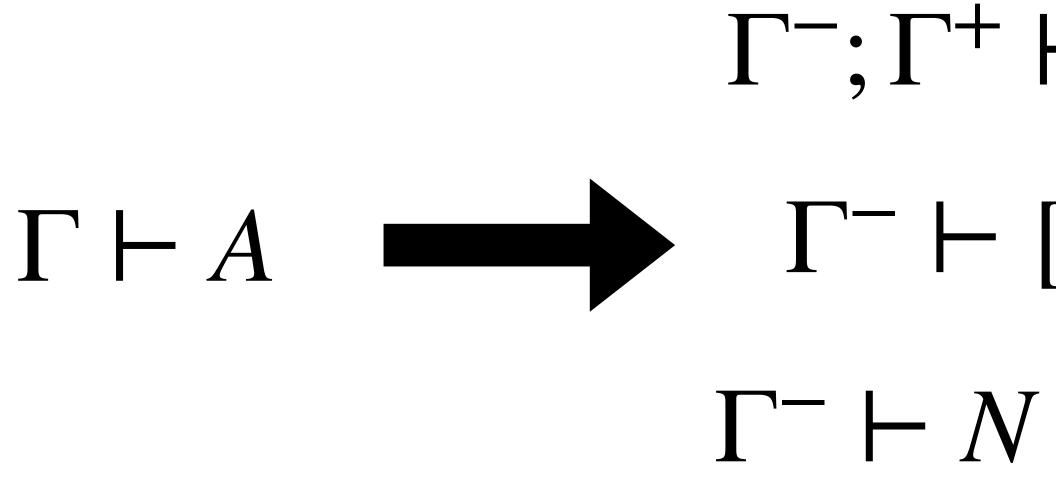
- Distinguish formulas (A) by polarity: positive (P, Q) and negative (N, O). Intuitionistic sequent calculus: non-backtracking left rules (+) vs. right rules (-) Γ^{-} : $\Gamma^{+} \vdash A$ inversion (non-backtracking rules) right focus (backtracking right rules)



 $\Gamma^- \vdash N > P$ left focus (backtracking left rules)



- Distinguish formulas (A) by polarity: positive (P, Q) and negative (N, O). Intuitionistic sequent calculus: non-backtracking left rules (+) vs. right rules (-) Γ^{-} : $\Gamma^{+} \vdash A$ inversion (non-backtracking rules) right focus (backtracking right rules) $\Gamma^- \vdash N > P$ left focus (backtracking left rules)



• Focalization: complete for classical linear logic (Andreoli, 1992), intuitionistic logic (Liang & Miller, 2009), etc.



focused intuitionistic logic

call-by-push-value

	focused intuitionistic logic	
t \(D_0_0	positive	
types		

call-by-push-value

value

	focused intuitionistic logic	call-by-push-value
tunan	positive	value
types	negative	computation

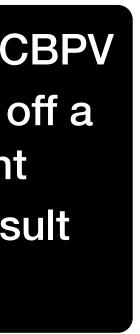
	focused intuitionistic logic	call-by-push-value
tuneo	positive	value
types	negative	computation
	inversion	computation
terms*		

	focused intuitionistic logic	call-by-push-value
	positive	value
types	negative	computation
	inversion	computation
terms*		

	focused intuitionistic logic	call-by-push-value
tupoo	positive	value
types	negative	computation
	inversion	computation
terms*	right focus	value

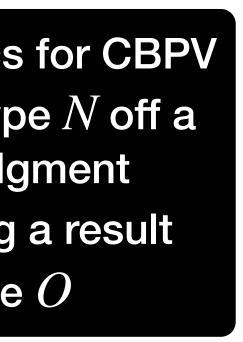
	focused intuitionistic logic	call-by-push-value
tunoo	positive	value
types	negative	computation
	inversion	computation
terms*	right focus	value
	weak left focus	stack

	focused intuitionistic logic	call-by-push-value	
t ura a a	positive	value	
types	negative	computation	
	inversion	computation	
terms*	right focus	value	The CK-machine semantics plays a computation of type
	weak left focus	stack	stack typed by the judgr $\Gamma^+ \vdash N \gg O$, producing a
			computation of type



	focused intuitionistic logic	call-by-push-value	
tuna	positive	value	
types	negative	computation	
	inversion	computation	
terms*	right focus	value	The CK-machine semantics plays a computation of typ
	weak left focus	stack	stack typed by the judg $\Gamma^+ \vdash N \gg O$, producing
reduction	focalization of CBPV term is f Zdancew		computation of type

*opposing contextual structure (insightful comments on Krishnaswami, 2018)

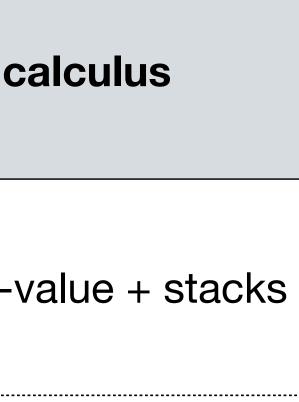


O

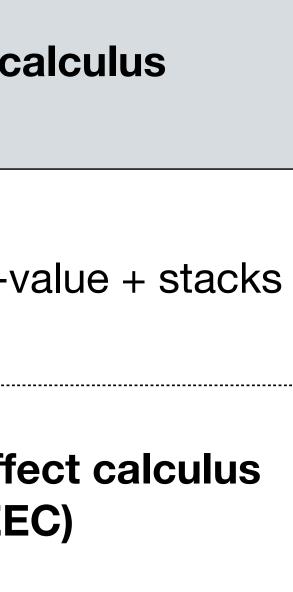
focusing effect ca



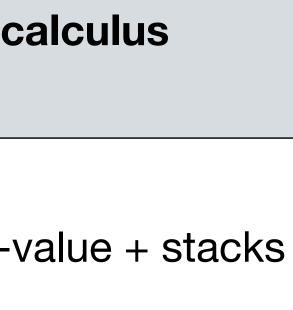
focusing	effect c
(weakly) focused intuitionistic logic	call-by-push-v



focusing	effect c
(weakly) focused intuitionistic logic	call-by-push-v
? (this talk, kind of)	enriched effe (EE



focusing	effect c
(weakly) focused intuitionistic logic	call-by-push-v
? (this talk, kind of)	enriched effe (EE



fect calculus EC)



Issue: EEC disagrees with focusing-theoretic polarity

focusing	effect c
(weakly) focused intuitionistic logic	call-by-push-v
? (this talk, kind of)	enriched effe
	(EE



value + stacks

fect calculus EC)

li (Curien et al., sumes EEC) Issue: EEC disagrees with focusing-theoretic polarity





Another Quick Introduction

 Observation: a stack constitutes a linear consumption of the input computation



Another Quick Introduction

- Observation: a stack constitutes a *linear* consumption of the input computation
- Idea: internalize stack judgment $N \gg O$ as the value type $N \multimap O$



Another Quick Introduction

- Observation: a stack constitutes a *linear* consumption of the input computation
- Idea: internalize stack judgment $N \gg O$ as the value type $N \multimap O$
 - Left rule internalizes CK-semantics, "looks like" function application



Another Quick Introduction

- Observation: a stack constitutes a *linear* consumption of the input computation
- Idea: internalize stack judgment $N \gg O$ as the value type $N \multimap O$
- Problem: function types shouldn't be positive (?)

Left rule internalizes CK-semantics, "looks like" function application

EEC

Another Quick Introduction

- Observation: a stack constitutes a *linear* consumption of the input computation
- Idea: internalize stack judgment $N \gg O$ as the value type $N \multimap O$
 - Left rule internalizes CK-semantics, "looks like" function application
- Problem: function types shouldn't be positive (?)
- Other EEC connectives have similar problems \Rightarrow different notion of polarity?

Curien & Munch-Maccagnoni, 2010; M-M & Scherer, 2013)

• Zeilberger, 2010: an inversion term is a continuation that consumes a focus term (also

- Zeilberger, 2010: an inversion term is a continuation that consumes a focus term (also Curien & Munch-Maccagnoni, 2010; M-M & Scherer, 2013)
- Formally, assume that [P] and N > P are defined. Then, where (\Rightarrow) is the meta arrow, let:

- Zeilberger, 2010: an inversion term is a continuation that consumes a focus term (also Curien & Munch-Maccagnoni, 2010; M-M & Scherer, 2013)
- Formally, assume that [P] and N > P are defined. Then, where (\Rightarrow) is the meta arrow, let:
 - Left inversion: $P', ..., Q' \vdash P \triangleq [P'] \Rightarrow ... \Rightarrow [Q'] \Rightarrow [P]$

- Zeilberger, 2010: an inversion term is a continuation that consumes a focus term (also Curien & Munch-Maccagnoni, 2010; M-M & Scherer, 2013)
- Formally, assume that [P] and N > P are defined. Then, where (\Rightarrow) is the meta arrow, let:
 - Left inversion: $P', \dots, Q' \vdash P \triangleq [P'] \Rightarrow \dots \Rightarrow [Q'] \Rightarrow [P]$
 - Right inversion: $\Gamma^+ \vdash N \triangleq$ for all $P : N > P \Rightarrow \Gamma^+ \vdash [P]$

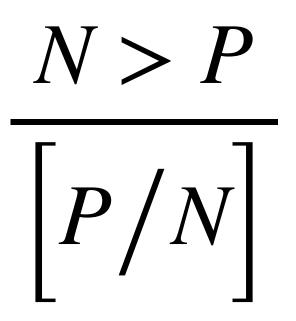
- Zeilberger, 2010: an inversion term is a continuation that consumes a focus term (also Curien & Munch-Maccagnoni, 2010; M-M & Scherer, 2013)
- Formally, assume that [P] and N > P are defined. Then, where (\Rightarrow) is the meta arrow, let:
 - Left inversion: $P', ..., Q' \vdash P \triangleq [P'] \Rightarrow ... \Rightarrow [Q'] \Rightarrow [P]$
 - Right inversion: $\Gamma^+ \vdash N \triangleq$ for all $P : N > P \Rightarrow \Gamma^+ \vdash [P]$
- Why bother?

- Zeilberger, 2010: an inversion term is a continuation that consumes a focus term (also Curien & Munch-Maccagnoni, 2010; M-M & Scherer, 2013)
- Formally, assume that [P] and N > P are defined. Then, where (\Rightarrow) is the meta arrow, let:
 - Left inversion: $P', ..., Q' \vdash P \triangleq [P'] \Rightarrow ... \Rightarrow [Q'] \Rightarrow [P]$
 - Right inversion: $\Gamma^+ \vdash N \triangleq$ for all $P : N > P \Rightarrow \Gamma^+ \vdash [P]$
- Why bother?
 - Polarity arises from value-continuation distinction, not rule (non)backtracking

- Zeilberger, 2010: an inversion term is a continuation that consumes a focus term (also Curien & Munch-Maccagnoni, 2010; M-M & Scherer, 2013)
- Formally, assume that [P] and N > P are defined. Then, where (\Rightarrow) is the meta arrow, let:
 - Left inversion: $P', ..., Q' \vdash P \triangleq [P'] \Rightarrow ... \Rightarrow [Q'] \Rightarrow [P]$
 - Right inversion: $\Gamma^+ \vdash N \triangleq$ for all $P : N > P \Rightarrow \Gamma^+ \vdash [P]$
- Why bother?
 - Polarity arises from value-continuation distinction, not rule (non)backtracking
 - However, defunctionalization recovers (first-order, CPS'd) focusing see Zeilberger, 2011; M-M's thesis, ch. 3

The Solution Type Structure of Extended Logic of DC \simeq EEC

The Solution Type Structure of Extended Logic of DC \simeq EEC



• Zeilberger, 2010: right focus internalizing left focus \Rightarrow positive type of stacks

The Solution Type Structure of Extended Logic of DC \simeq EEC

- Zeilberger, 2010: right focus internalizing left focus \Rightarrow positive type of stacks
- Our contribution: linear functions, other EEC connectives

$$N > P \qquad \text{for all}$$

$$\left[P/N \right]$$

• Term correspondence?

- Term correspondence?
 - Weak left focus is not primitive b/ would be circularly defined

- Term correspondence?
 - would be circularly defined
 - So, linear functions are stack transformers (Barrett et al., 2023)

- Term correspondence?
 - would be circularly defined
 - So, linear functions are stack transformers (Barrett et al., 2023)
- Five cut, two identity rules: all admissible (easily)

- Term correspondence?
 - would be circularly defined
 - So, linear functions are stack transformers (Barrett et al., 2023)
- Five cut, two identity rules: all admissible (easily)
- Categorical semantics? More on that later...

to:

• "Straightforward" to generalize to polymorphic, dependent types; corresponds

- to:
 - 2020; Jiang et al. 2025)

• "Straightforward" to generalize to polymorphic, dependent types; corresponds

Polymorphic CBPV/EEC (Møgelberg & Simpson, 2009; Rioux & Zdancewic,

- to:
 - 2020; Jiang et al. 2025)
 - Focused linear dependent types, analogous to eMLTT (Ahman, 2017)

• "Straightforward" to generalize to polymorphic, dependent types; corresponds

Polymorphic CBPV/EEC (Møgelberg & Simpson, 2009; Rioux & Zdancewic,

- "Straightforward" to generalize to polymorphic, dependent types; corresponds to:
 - Polymorphic CBPV/EEC (Møgelberg & Simpson, 2009; Rioux & Zdancewic, 2020; Jiang et al. 2025)
 - Focused linear dependent types, analogous to eMLTT (Ahman, 2017)
- "Straightforward" to defunctionalize the calculus

- "Straightforward" to generalize to polymorphic, dependent types; corresponds to:
 - Polymorphic CBPV/EEC (Møgelberg & Simpson, 2009; Rioux & Zdancewic, 2020; Jiang et al. 2025)
 - Focused linear dependent types, analogous to eMLTT (Ahman, 2017)
- "Straightforward" to defunctionalize the calculus
 - Would appear closer to EEC, L-calculi \Rightarrow categorical semantics by translation?



- "Straightforward" to generalize to polymorphic, dependent types; corresponds to:
 - Polymorphic CBPV/EEC (Møgelberg & Simpson, 2009; Rioux & Zdancewic, 2020; Jiang et al. 2025)
 - Focused linear dependent types, analogous to eMLTT (Ahman, 2017)
- "Straightforward" to defunctionalize the calculus
 - Would appear closer to EEC, L-calculi \Rightarrow categorical semantics by translation?
- Working on relationship between stacks and linear lenses



Thanks <u>ssomayya@alumni.cmu.edu</u>

