THE SEMANTIC ANALYSIS OF ADVANCED PROGRAMMING LANGUAGES

Harley Eades III
Computer Science
University of Iowa

Can Software be Trusted?

Can Research be Trusted?

Hope for the Future

- How do we fix these problems?
  - Must go down to the foundation!
- Programming languages (PL) must be extended to support verification of software correctness.

Typed Functional PL

- Rich type systems allow for the encoding of correctness properties.
- These correctness properties then imply correctness of the software being implemented.
- This is where my research comes in.

Overview

- Design
- Freedom of Speech
- Separation of Proof from Program (Sep3)
- Dualized Type Theory (DTT)

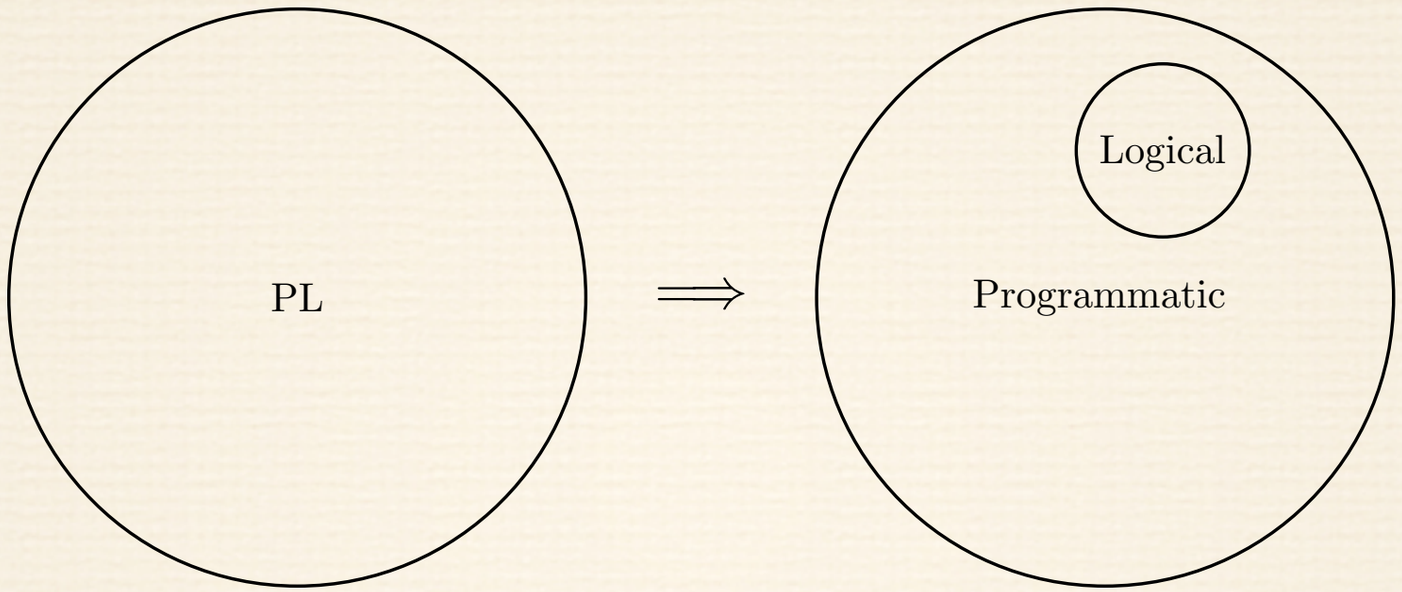
Overview

- Analysis
  - Basic Syntactic Analysis
    - Freedom of Speech
    - DTT
- Normalization by Hereditary Substitution
  - Stratified System F (SSF) and Beyond
  - The λΔ-calculus

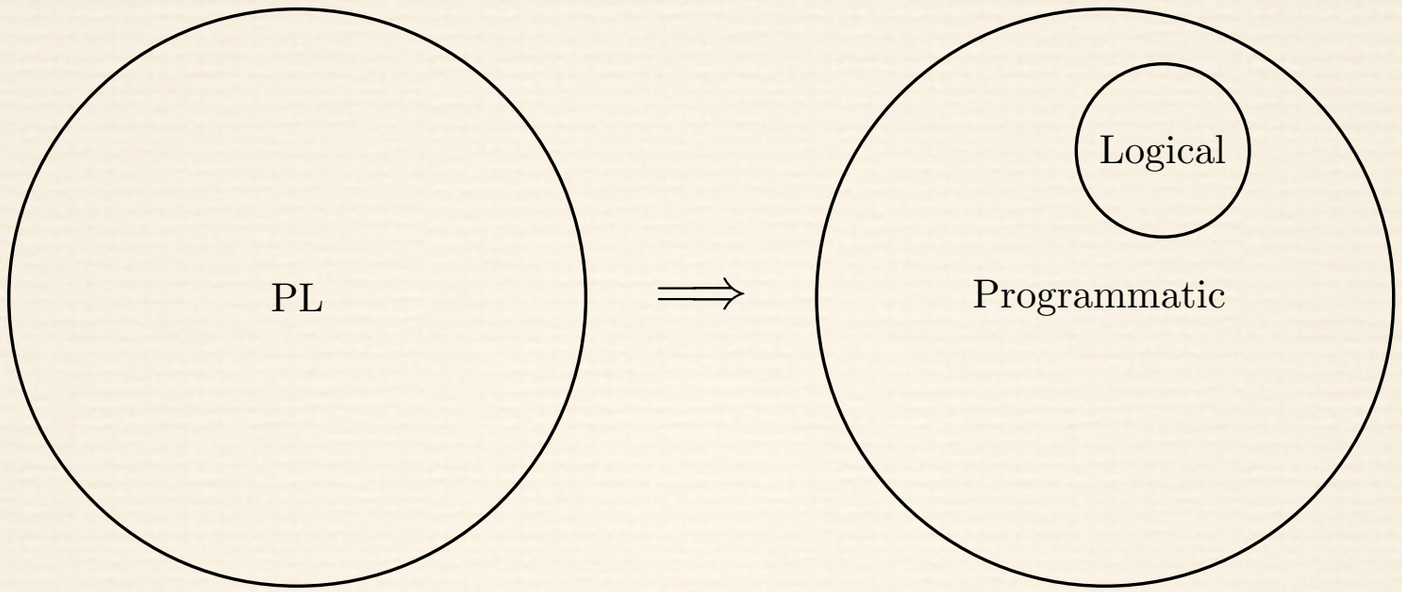
Detailed Background

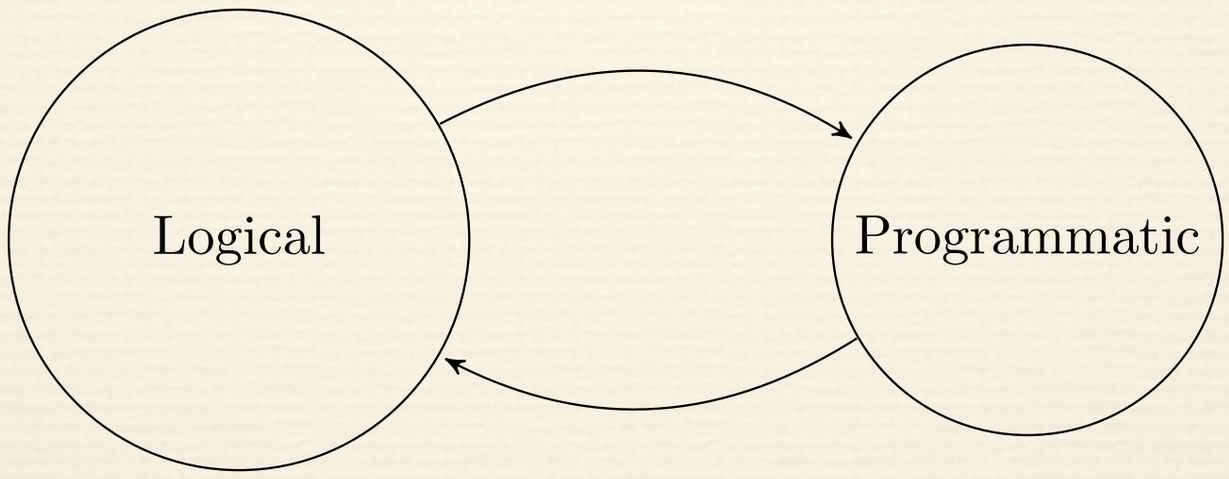
One-stop shop for newcomers.

Freedom of Speech



Freedom of Speech

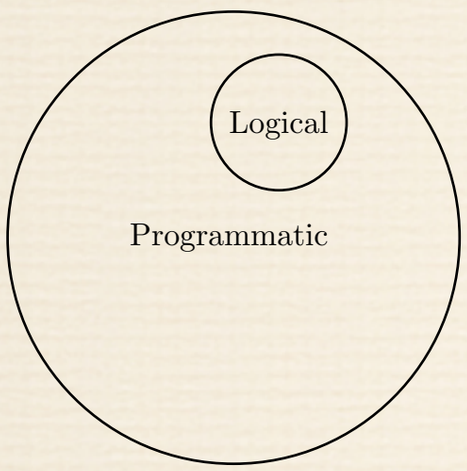




freedom of speech property

Freedom of Speech

$\Gamma \vdash^P e : e'$



$\Gamma \vdash^L e : e'$

$$\frac{\Gamma \vdash^\theta e_1 : \mathsf{Type} \qquad \Gamma, x :^\theta e_1 \vdash^{\theta'} e : e_2}{\Gamma \vdash^{\theta'} \lambda x \,.\, e : (x :^\theta e_1)^+ \to e_2} \textsc{Lam}$$

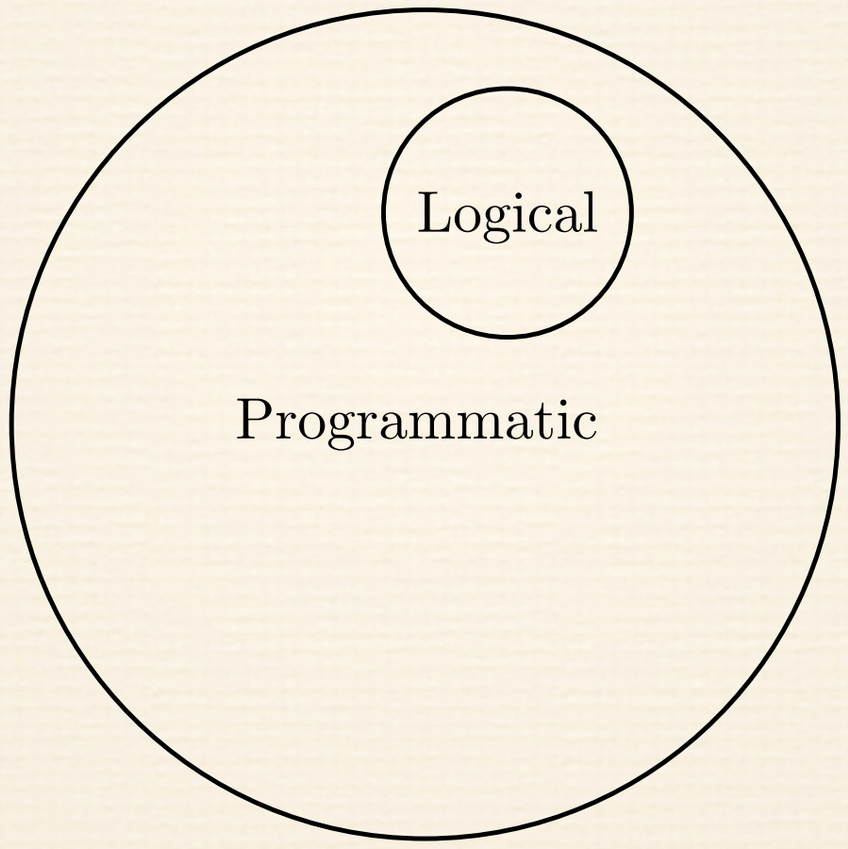
$$\frac{e \downarrow e' \qquad \Gamma \vdash^{\theta_1} e : e_1 \qquad \Gamma \vdash^{\theta_2} e' : e_2}{\Gamma \vdash^L \mathsf{join} : e = e'} \textsc{join}$$

$$\frac{\Gamma \vdash^\theta [e_1'/x]e_2 : \mathsf{Type} \qquad \Gamma \vdash^\theta e : [e_1/x]e_2 \qquad \Gamma \vdash^L e' : e_1 = e_1'}{\Gamma \vdash^\theta e : [e_1'/x]e_2} \textsc{Conv}$$

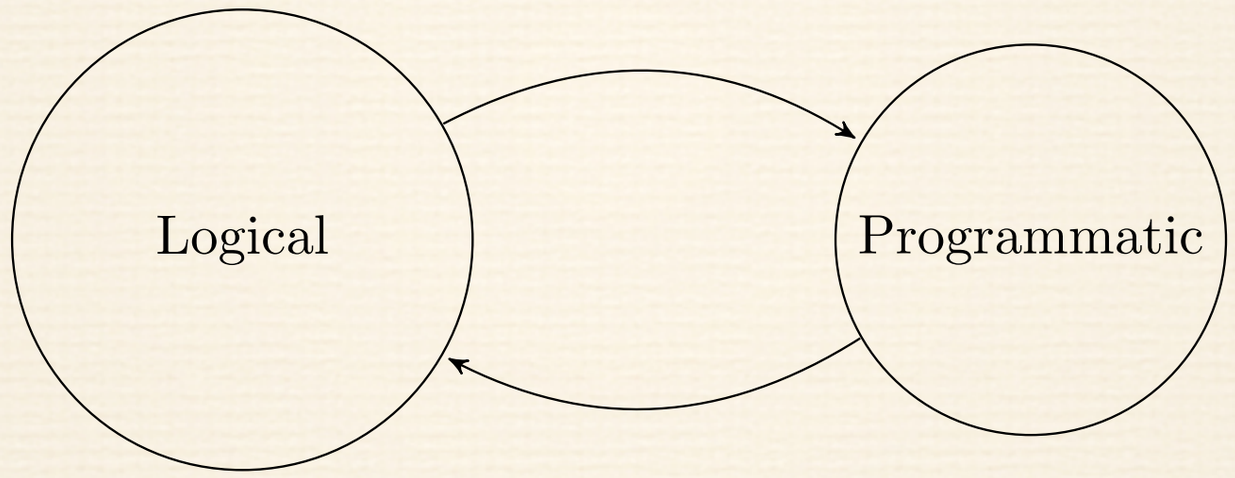
Freedom of Speech

- Type Preservation
- Weak normalization.
  - Depends on type preservation.
  - A deep interpretation of logical programs while a shallow interpretation of all other programs.

Separation of Proof from Program (Sep3)



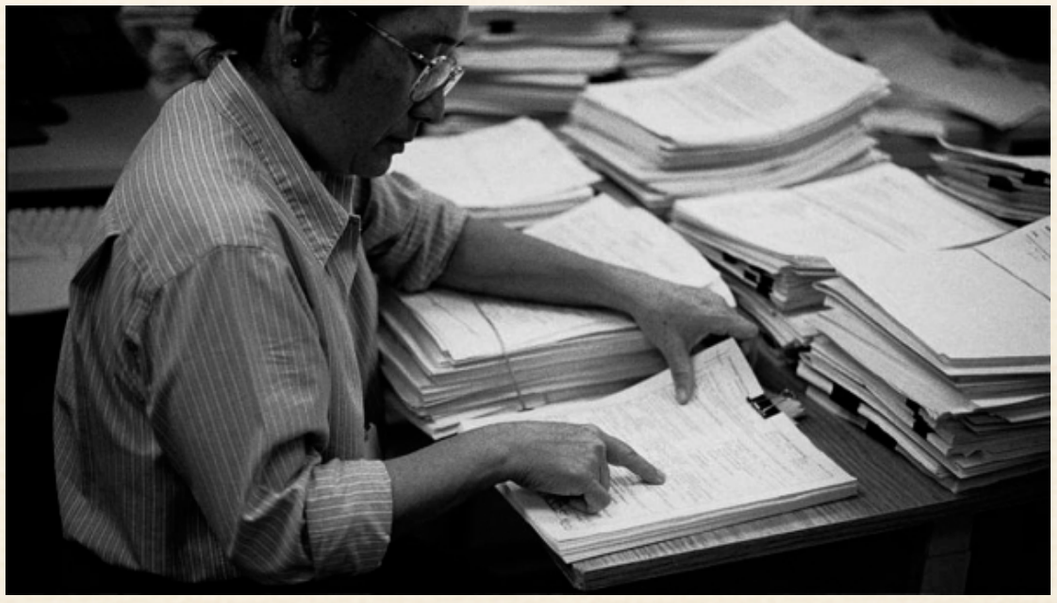
Separation of Proof from Program (Sep3)



Separation of Proof from Program (Sep3)

- Lifts the value restrictions found in Freedom of Speech.
  - Due to the strict separation of the logical fragment and the programmatic fragment.
- Extensions: Datatypes and a higher-order predicative logic (polymorphism).

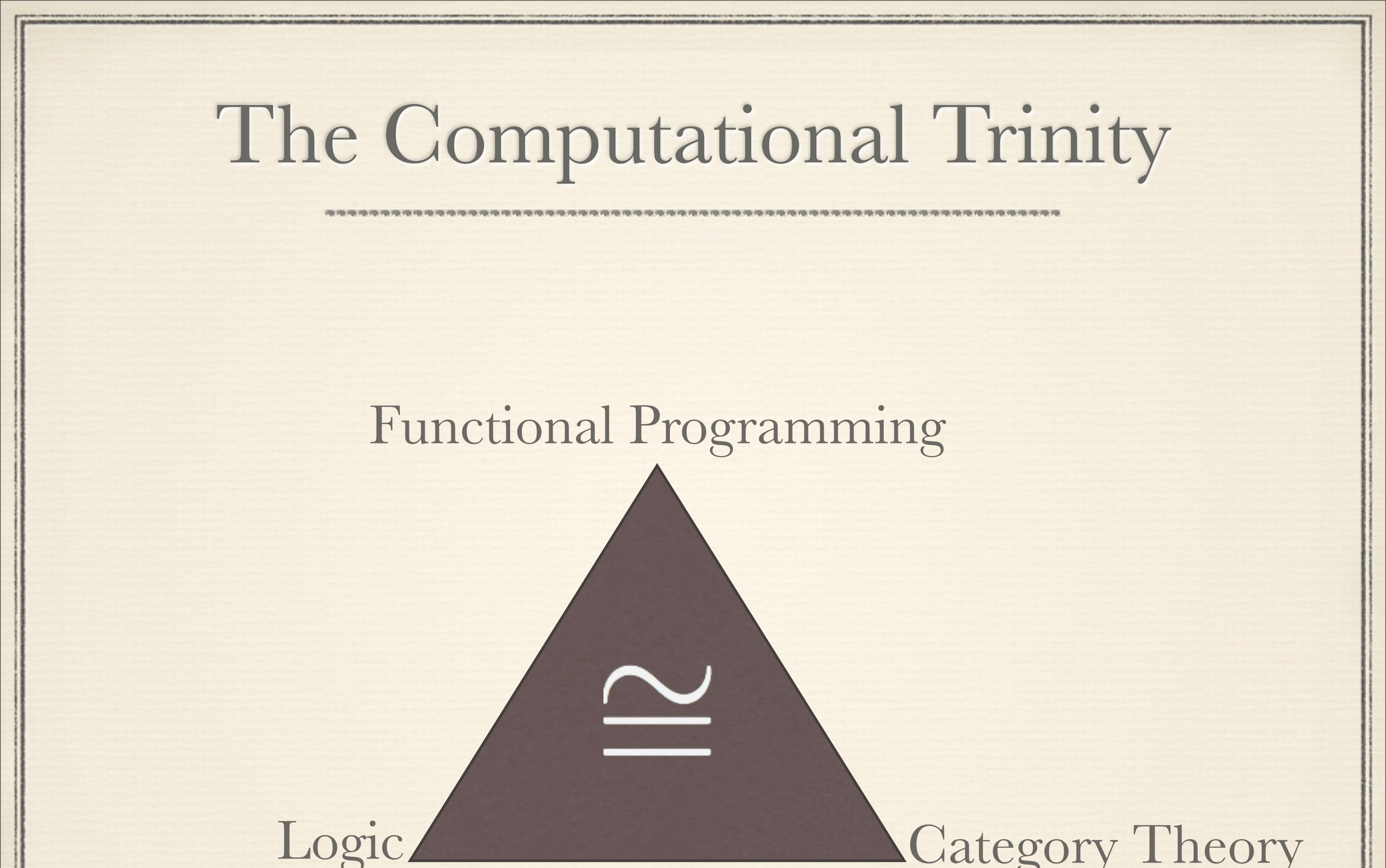
Motivation



Concentrate on the Logical Fragment

- How can we add coinduction?
- How about mixed induction-coinduction?
- Can we add both of these in a consistent and type safe way?

The Computational Trinity



Dualized Intuitionistic Logic

- Based on bi-intuitionistic logic.
  - Perfect duality.
  - A simplification of Pinto and Uustalu's system L.
    - $n_1 : A_1, \ldots, n_i : A_i \vdash_G m_1 : B_1, \ldots, m_j : B_j$
    - Converted sequents to single-sided sequents.
    - A new dualized syntax.

Dualized Intuitionistic Logic

$$\frac{G \vdash n \preccurlyeq^*_p n'}{G; \Gamma, p\ A \mathbin{@} n, \Gamma' \vdash p\ A \mathbin{@} n'}\ \textsc{ax} \qquad \frac{}{G; \Gamma \vdash p\ \langle p \rangle \mathbin{@} n}\ \textsc{unit}$$

$$\frac{G; \Gamma \vdash p\ A \mathbin{@} n \qquad G; \Gamma \vdash p\ B \mathbin{@} n}{G; \Gamma \vdash p\ (A \wedge_p B) \mathbin{@} n}\ \textsc{and}$$

$$\frac{G; \Gamma \vdash p\ A_d \mathbin{@} n}{G; \Gamma \vdash p\ (A_1 \wedge_{\bar{p}} A_2) \mathbin{@} n}\ \textsc{and}\textsc{Bar}$$

$$\frac{\begin{array}{l} n' \notin |G|, |\Gamma| \\ (G, n \preccurlyeq^p n'); \Gamma, p\ A \mathbin{@} n' \vdash p\ B \mathbin{@} n' \end{array}}{G; \Gamma \vdash p\ (A \to_p B) \mathbin{@} n}\ \textsc{imp}$$

$$\frac{\begin{array}{l} G \vdash n \preccurlyeq^*_{\bar{p}} n' \\ G; \Gamma \vdash \bar{p}\ A \mathbin{@} n' \qquad G; \Gamma \vdash p\ B \mathbin{@} n' \end{array}}{G; \Gamma \vdash p\ (A \to_{\bar{p}} B) \mathbin{@} n}\ \textsc{imp}\textsc{Bar}$$

$$\frac{G; \Gamma, \bar{p}\ A \mathbin{@} n \vdash +\ B \mathbin{@} n' \qquad G; \Gamma, \bar{p}\ A \mathbin{@} n \vdash -\ B \mathbin{@} n'}{G; \Gamma \vdash p\ A \mathbin{@} n}\ \textsc{cut}$$

Dualized Type Theory

$$\frac{G \vdash n \preccurlyeq^*_p n'}{G; \Gamma, x : p\, A \mathbin{@} n, \Gamma' \vdash x : p\, A \mathbin{@} n'} \textsc{Ax} \qquad \frac{}{G; \Gamma \vdash \mathbf{triv} : p\, \langle p \rangle \mathbin{@} n} \textsc{Unit}$$

$$\frac{G; \Gamma \vdash t_1 : p\, A \mathbin{@} n \qquad G; \Gamma \vdash t_2 : p\, B \mathbin{@} n}{G; \Gamma \vdash (t_1, t_2) : p\, (A \wedge_p B) \mathbin{@} n} \textsc{And}$$

$$\frac{G; \Gamma \vdash t : p\, A_d \mathbin{@} n}{G; \Gamma \vdash \mathbf{in}_d\, t : p\, (A_1 \wedge_{\bar{p}} A_2) \mathbin{@} n} \textsc{AndBar}$$

$$\frac{\begin{array}{l} n' \notin |G|, |\Gamma| \\ (G, n \preccurlyeq^p n'); \Gamma, x : p\, A \mathbin{@} n' \vdash t : p\, B \mathbin{@} n' \end{array}}{G; \Gamma \vdash \lambda x.t : p\, (A \to_p B) \mathbin{@} n} \textsc{Imp}$$

$$\frac{\begin{array}{l} G \vdash n \preccurlyeq^*_{\bar{p}} n' \\ G; \Gamma \vdash t_1 : \bar{p}\, A \mathbin{@} n' \qquad G; \Gamma \vdash t_2 : p\, B \mathbin{@} n' \end{array}}{G; \Gamma \vdash \langle t_1, t_2 \rangle : p\, (A \to_{\bar{p}} B) \mathbin{@} n} \textsc{ImpBar}$$

$$\frac{\begin{array}{c} G; \Gamma, x : \bar{p}\, A \mathbin{@} n \vdash t_1 : + B \mathbin{@} n' \\ G; \Gamma, x : \bar{p}\, A \mathbin{@} n \vdash t_2 : - B \mathbin{@} n' \end{array}}{G; \Gamma \vdash \nu\, x.t_1 \bullet t_2 : p\, A \mathbin{@} n} \textsc{Cut}$$

Dualized Type Theory

- Metatheory:
  - DIL:
    - Consistency with respect to Rauszer's Kripke semantics -- proof in Agda.
    - Completeness by reduction to L.
  - DTT:
    - Type preservation
    - Strong normalization.

Analysis of PLs

Can we trust our verification?

Hereditary Substitution

- Syntax: $[t/x]^A\, t$
- Usual termination order: (A, t')
- Like ordinary capture-avoiding substitution.
- Except, if the substitution introduces a redex, then that redex is recursively reduced.
- Example:

$$[\lambda z : \mathrm{b}.z/x]^{\mathrm{b}\to\mathrm{b}}(x\, y)(\approx ((\lambda z : \mathrm{b}.z)\, y \approx [y/z]^{\mathrm{b}}\, z) = y$$

Why Hereditary Substitution?

Provides a directly defined substitution which preserves normal forms.

Proof by Hereditary Substitution

- Define an ordering on types.
- Define the hereditary substitution function.
- Prove the properties of hereditary substitution.

Properties

- Total and Type Preservation:

  Suppose $\Gamma \vdash t : T$ and $\Gamma, x : T, \Gamma' \vdash t' : T'$. Then there exists a term t'', such that, $[t/x]^T t' = t''$ and $\Gamma, \Gamma' \vdash t'' : T'$.

- Redex Preserving:

  If $\Gamma \vdash t : T$, $\Gamma, x : T, \Gamma' \vdash t' : T'$, then $|rset(t', t)| \geq |rset([t/x]^T t')|$.

- Normality Preserving:

  If $\Gamma \vdash n : T$ and $\Gamma, x : T \vdash n' : T'$ then there exists a normal term n'' such that $[n/x]^T n' = n''$.

Properties

- Soundness with Respect to Reduction:

$$\text{If } \Gamma \vdash t : T \text{ and } \Gamma, x : T, \Gamma' \vdash t' : T' \text{ then } [t/x]t' \rightsquigarrow^* [t/x]^T t'.$$

Interpretation of Types

Definition. *The interpretation of types* $[\![T]\!]_\Gamma$ *is defined by:*

$$n \in [\![T]\!]_\Gamma \quad \Longleftrightarrow \quad \Gamma \vdash n : T$$

We extend this definition to non-normal terms t *in the following way:*

$$t \in [\![T]\!]_\Gamma \quad \Longleftrightarrow \quad \exists n.t \rightsquigarrow^* n \in [\![T]\!]_\Gamma$$

Interpretation of Types

Lemma (Hereditary Substitution for the Interpretation of Types)**.** *If* $n \in [\![T]\!]_\Gamma$ *and* $n' \in [\![T']\!]_{\Gamma,x:T,\Gamma'}$*, then* $[n/x]^T n' \in [\![T']\!]_{\Gamma,\Gamma'}$*.*

Proof. We know by totality and type preservation that there exists a term s such that $[n/x]^T n' = s$ and $\Gamma, \Gamma' \vdash s : T'$, and by normality preservation s is normal. Therefore, $s \in [\![T']\!]_{\Gamma,\Gamma'}$. □

Concluding Normalization

Theorem (Type Soundness). *If* $\Gamma \vdash t : T$ *then* $t \in [\![T]\!]_\Gamma$.

Corollary (Normalization). *If* $\Gamma \vdash t : T$ *then there exists a term* n *such that* $t \rightsquigarrow^* n$.

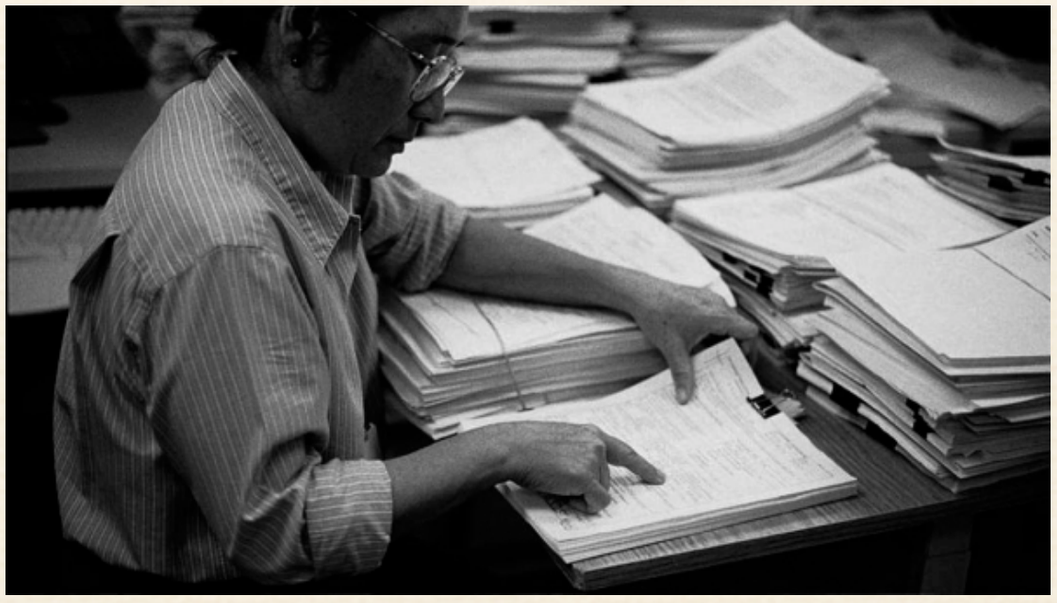
Stratified System F and Beyond

- Stratified System F (SSF)
- SSF with sum types and commuting conversions.
- Dependent SSF.
- SSFω

The λΔ-Calculus

- A classical type theory proposed by Jacob Rehöf.
- Essentially STLC with continuations.
- First classical type theory proved weakly normalizing using hereditary substitution.

Conclusion



Acknowledgements

- My wife, Jenny Eades.
- For all her hard work!

Acknowledgements

- Aaron Stump.
  - Introducing me to type theory.
  - Taking a chance on me.

Acknowledgements

- My committee:
  - Stephanie Weirich,
  - Cesare Tinelli,
  - Kasturi Varadarajan, and
  - Greg Landini.

Most Common Words in Emails with Aaron