Explicit Substitutions Part 2: Explicit Substitutions

slides: https://github.com/donovancrichton/Talks

Donovan Crichton

August 2025

The goal of this talk

- Introduce explicit substitution extensions to the STLC.

The goal of this talk

- Introduce explicit substitution extensions to the STLC.
- Provide examples on what the new substitutions mean.
  between types, terms, contexts, and substitutions.

The goal of this talk

- Introduce explicit substitution extensions to the STLC.
- Provide examples on what the new substitutions mean.
- Present an equational theory that determines relationships between types, terms, contexts, and substitutions.

The goal of this talk

- Introduce explicit substitution extensions to the STLC.
- Provide examples on what the new substitutions mean.
- Present an equational theory that determines relationships between types, terms, contexts, and substitutions.
- ~~Scale up to a dependent type theory~~

Untyped Lambda Calculus

$$
\begin{aligned}
V &::= x, y, z, \ ... \\
M, N &::= V && \text{Variable.} \\
&\mid M\ N && \text{Application.} \\
&\mid \lambda V.M && \text{Abstraction.}
\end{aligned}
$$

STLC Grammar

The Simply Typed Lambda Calculus: Grammar

$$V ::= \; x, y, z, \; \ldots \qquad \text{Variable Set}$$

STLC Grammar

The Simply Typed Lambda Calculus: Grammar

$$
\begin{aligned}
V &::= x, y, z, \ \ldots && \text{Variable Set} \\
\alpha, \beta &::= * \mid \alpha \to \beta && \text{Types}
\end{aligned}
$$

STLC Grammar

The Simply Typed Lambda Calculus: Grammar

$V ::=$	$x, y, z, \ ...$	Variable Set
$\alpha, \beta ::=$	$* \mid \alpha \to \beta$	Types
$M, N ::=$	$V \mid M\ N \mid \lambda(V : \alpha).M$	Terms

STLC Grammar

The Simply Typed Lambda Calculus: Grammar

$$
\begin{aligned}
V &::= \; x, y, z, \; \ldots && \text{Variable Set}\\
\alpha, \beta &::= \; * \mid \alpha \to \beta && \text{Types}\\
M, N &::= V \;\mid M\; N \mid \lambda(V : \alpha).M && \text{Terms}\\
\Gamma &::= \; \diamond \mid \Gamma, x_m : \alpha && \text{Contexts}
\end{aligned}
$$

STLC Grammar

The Simply Typed Lambda Calculus: Grammar

$$
\begin{aligned}
V &::= \; x, y, z, \; ... && \text{Variable Set} \\
\alpha, \beta &::= \; * \mid \alpha \to \beta && \text{Types} \\
M, N &::= V \; \mid M \; N \mid \lambda(V : \alpha).M && \text{Terms} \\
\Gamma &::= \; \diamond \mid \Gamma, x_m : \alpha && \text{Contexts}
\end{aligned}
$$

Why $x_m : \alpha$ in the Contexts set, and not $V : \alpha$?

STLC Grammar

The Simply Typed Lambda Calculus: Grammar

$$
\begin{aligned}
V &::= \ x, y, z, \ \ldots && \text{Variable Set} \\
\alpha, \beta &::= \ * \mid \alpha \to \beta && \text{Types} \\
M, N &::= V \ \mid M \ N \mid \lambda(V : \alpha).M && \text{Terms} \\
\Gamma &::= \ \diamond \mid \Gamma, x_m : \alpha && \text{Contexts}
\end{aligned}
$$

Why $x_m : \alpha$ in the Contexts set, and not $V : \alpha$?

x_m represents a variable in our meta language

STLC Rules

STLC: Typing Rules

$$\frac{\Gamma \vdash \alpha \ \mathsf{Type} \qquad \Gamma \vdash \beta \ \mathsf{Type}}{\Gamma \vdash \alpha \to \beta \ \mathsf{Type}} (\to\text{-}\mathit{Form})$$

STLC Rules

STLC: Typing Rules

$$\frac{\Gamma \vdash \alpha \text{ Type} \qquad \Gamma \vdash \beta \text{ Type}}{\Gamma \vdash \alpha \to \beta \text{ Type}} (\to\textit{-Form})$$

$$\frac{\Gamma \vdash \alpha \to \beta \text{ Type} \qquad \Gamma, x_m : \alpha \vdash M : \beta}{\Gamma \vdash \lambda(x : \alpha).M : \alpha \to \beta} (\to\textit{-Intro})$$

STLC Rules

STLC: Typing Rules

$$\frac{\Gamma \vdash \alpha \text{ Type} \qquad \Gamma \vdash \beta \text{ Type}}{\Gamma \vdash \alpha \to \beta \text{ Type}}(\to\textit{-Form})$$

$$\frac{\Gamma \vdash \alpha \to \beta \text{ Type} \qquad \Gamma, x_m : \alpha \vdash M : \beta}{\Gamma \vdash \lambda(x : \alpha).M : \alpha \to \beta}(\to\textit{-Intro})$$

$$\frac{\Gamma \vdash \alpha \to \beta \text{ Type} \qquad \Gamma \vdash M : \alpha \to \beta \qquad \Gamma \vdash N : \alpha}{\Gamma \vdash M\ N : \beta}(\to\textit{-Elim})$$

Explicit Substitution Grammar

STLC with Explicit Substitutions: Grammar

$$\alpha, \beta ::= \ * \mid \alpha \to \beta \qquad \text{Types}$$

Explicit Substitution Grammar

STLC with Explicit Substitutions: Grammar

$\alpha, \beta ::= * \mid \alpha \to \beta$	Types
$M, N ::= 1 \mid M\ N \mid \lambda\alpha.M \mid M[\gamma]$	Terms

Explicit Substitution Grammar

STLC with Explicit Substitutions: Grammar

$$\begin{array}{rl} \alpha, \beta ::= \ * \mid \alpha \to \beta & \text{Types} \\ M, N ::= 1 \mid M\ N \mid \lambda\alpha.M \mid M[\gamma] & \text{Terms} \\ \gamma, \delta ::= \mathit{id} \mid \uparrow \mid \gamma, M : \alpha \mid \gamma \circ \delta & \text{Substitutions} \end{array}$$

Explicit Substitution Grammar

STLC with Explicit Substitutions: Grammar

$$\begin{aligned}
\alpha, \beta &::= * \mid \alpha \to \beta && \text{Types} \\
M, N &::= 1 \mid M\ N \mid \lambda\alpha.M \mid M[\gamma] && \text{Terms} \\
\gamma, \delta &::= id \mid \uparrow \mid \gamma, M : \alpha \mid \gamma \circ \delta && \text{Substitutions} \\
\Gamma &::= \diamond \mid \Gamma, \alpha && \text{Contexts}
\end{aligned}$$

Explicit Subs: Terms

Explicit Substitution Terms

$$M, N ::= 1 \mid M\ N \mid \lambda\alpha.M \mid M[\gamma] \qquad \text{Terms}$$

Explicit Subs: Terms

Explicit Substitution Terms

$$M, N ::= 1 \mid M\ N \mid \lambda\alpha.M \mid M[\gamma] \qquad \text{Terms}$$

Intuition

- $M\ N$ is our standard application.

Explicit Subs: Terms

Explicit Substitution Terms

$$M, N ::= 1 \mid M\ N \mid \lambda\alpha.M \mid M[\gamma] \qquad \text{Terms}$$

Intuition

- $M\ N$ is our standard application.
- $\lambda\alpha.M$ abstraction refers to the type, but no longer the variable.

Explicit Subs: Terms

Explicit Substitution Terms

$$M, N ::= 1 \mid M\ N \mid \lambda\alpha.M \mid M[\gamma] \qquad \text{Terms}$$

Intuition

- $M\ N$ is our standard application.
- $\lambda\alpha.M$ abstraction refers to the type, but no longer the variable.
- 1 is the most recently bound De-Bruijn index variable.

Explicit Subs: Terms

Explicit Substitution Terms

$$M, N ::= 1 \mid M\ N \mid \lambda\alpha.M \mid M[\gamma] \qquad \text{Terms}$$

Intuition

- $M\ N$ is our standard application.
- $\lambda\alpha.M$ abstraction refers to the type, but no longer the variable.
- 1 is the most recently bound De-Bruijn index variable.
- $M[\gamma]$ is term M with substituion γ applied.

Explicit Subs: Subs

Explicit Substitutions: Subs

$$\gamma, \delta ::= \mathit{id} \mid \uparrow \mid \gamma, M : \alpha \mid \gamma \circ \delta \qquad \text{Substitutions}$$

Explicit Subs: Subs

Explicit Substitutions: Subs

$$\gamma, \delta ::= \mathit{id} \mid \uparrow \mid \gamma, M : \alpha \mid \gamma \circ \delta \qquad \text{Substitutions}$$

Intuition

- *id* is the identity substitution.

Explicit Subs: Subs

Explicit Substitutions: Subs

$$\gamma, \delta ::= \mathit{id} \mid \uparrow \mid \gamma, M : \alpha \mid \gamma \circ \delta \qquad \text{Substitutions}$$

Intuition

- *id* is the identity substitution, mapping the source context to itself.
- $\uparrow$ is the weakening or shifting substituion. We 'weaken' the context by one additional variable.

Explicit Subs: Subs

Explicit Substitutions: Subs

$$\gamma, \delta ::= \mathit{id} \mid \uparrow \mid \gamma, M : \alpha \mid \gamma \circ \delta \qquad \text{Substitutions}$$

Intuition

- *id* is the identity substitution, mapping the source context to itself.
- $\uparrow$ is the weakening or shifting substituion. We 'weaken' the context by one additional variable.
- γ, M is the substitution extension or snoc. We extend the substitution by a term.

Explicit Subs: Subs

Explicit Substitutions: Subs

$$\gamma, \delta ::= \mathit{id} \mid \uparrow \mid \gamma, M : \alpha \mid \gamma \circ \delta \qquad \text{Substitutions}$$

Intuition

- *id* is the identity substitution, mapping the source context to itself.
- $\uparrow$ is the weakening or shifting substituion. We 'weaken' the context by one additional variable.
- $\gamma, M : \alpha$ is the substitution extension or snoc. We extend the substitution by a term.
- $\gamma \circ \delta$ is the composition substitution where we compose two substitutions together

Example: Identity

$\diamond, \alpha, \beta, \theta \vdash \mathit{id} : \diamond, \alpha, \beta, \theta$

We move from source context to target context with no changes.

Example: Identity

$\diamond, \alpha, \beta, \theta \vdash \mathit{id} : \diamond, \alpha, \beta, \theta$

We move from source context to target context with no changes.

$$\begin{tikzcd} \diamond, \alpha, \beta, \theta \arrow[d, "\mathit{id}"] \\ \diamond, \alpha, \beta, \theta \end{tikzcd}$$

Example: Weakening (Shift)

$\diamond, \alpha, \beta, \theta \vdash \uparrow : \diamond, \alpha, \beta$

We add a new variable (weaken) the context, which shifts everything else up by one index.

Example: Weakening (Shift)

$\diamond, \alpha, \beta, \theta \vdash \uparrow : \diamond, \alpha, \beta$

We add a new variable (weaken) the context, which shifts everything else up by one index.

$$\begin{tikzcd} \diamond, \overset{2}{\alpha}, \overset{1}{\beta} \arrow[d, "\uparrow"'] \\ \diamond, \overset{3}{\alpha}, \overset{2}{\beta}, \overset{1}{\theta} \end{tikzcd}$$

Example: Extension (Snoc)

$\diamond, \alpha, \beta \vdash \ (\gamma, M : \theta) \ : \diamond, \alpha, \beta, \theta$

We extend substituion γ by a term $M : \theta$ and drop θ from our source context. This models the act of substituting in a term for a variable in context.

Example: Extension (Snoc)

$\diamond, \alpha, \beta \vdash (\gamma, M : \theta) : \diamond, \alpha, \beta, \theta$

We extend substituion γ by a term $M : \theta$ and drop θ from our source context. This models the act of substituting in a term for a variable in context.

$$\begin{tikzcd}
\diamond, \overset{3}{\alpha}, \overset{2}{\beta}, \overset{1}{\theta} \arrow[d, "{\gamma, M : \theta}"] \\
\diamond, \overset{2}{\alpha}, \overset{1}{\beta}
\end{tikzcd}$$

Example: Composition

Composition combines two substitutions

If $\Delta \vdash \gamma : \Gamma$ and $\Theta \vdash \delta : \Delta$
then $\Theta \vdash \gamma \circ \delta : \Gamma$

Example: Composition

Composition combines two substitutions

If $\Delta \vdash \gamma : \Gamma$ and $\Theta \vdash \delta : \Delta$
then $\Theta \vdash \gamma \circ \delta : \Gamma$

$$\begin{tikzcd} \Gamma \\ \Delta \arrow[u, "\gamma"'] \\ \Theta \arrow[u, "\delta"'] \end{tikzcd}$$

Read upward:
δ then γ

Example: Composition

Composition combines two substitutions

If $\Delta \vdash \gamma : \Gamma$ and $\Theta \vdash \delta : \Delta$
then $\Theta \vdash \gamma \circ \delta : \Gamma$

$$
\begin{tikzcd}
\Gamma & & \Gamma \\
\Delta \arrow[u, "\gamma"'] & = & \\
\Theta \arrow[u, "\delta"'] & & \Theta \arrow[uu, "\gamma \circ \delta"']
\end{tikzcd}
$$

Example: Composition

Example: Specific Composition

Consider $\diamond, \alpha, \beta \vdash \gamma : \diamond, \alpha$ and $\diamond, \alpha, \beta, \theta \vdash \delta : \diamond, \alpha, \beta$

$$\begin{tikzcd} \diamond, \alpha \\ \diamond, \alpha, \beta \arrow[u, "\gamma"] \\ \diamond, \alpha, \beta, \theta \arrow[u, "\delta"] \end{tikzcd}$$

Result: $\diamond, \alpha, \beta, \theta \vdash \gamma \circ \delta : \diamond, \alpha$

Reading the diagram

Start from $\diamond, \alpha, \beta, \theta$, apply δ to get $\diamond, \alpha, \beta$,
then apply γ to reach $\diamond, \alpha$

A Note On Notation: Composition and Context

A Note On Notation: Composition and Context

Two notations for composition exist in the literature:

- **Sequential (Abadi et al. original):** $\gamma \circ \delta$ means "first γ, then δ"
- **Categorical (most papers):** $\delta \circ \gamma$ means "first γ, then δ"
- In this talk: We use $\gamma \circ \delta$ meaning "first δ, then γ"

Why the difference?

The original paper used sequential notation matching evaluation order.
Modern presentations use categorical notation matching function composition.

A Note On Notation: Composition and Context

A Note On Notation: Composition and Context

Critical: Pay attention to context order in substitution typing!

In the judgment $\Gamma \vdash \gamma : \Delta$:

- Δ is the **source** context (domain)
- Γ is the **target** context (codomain)
- γ maps FROM Δ TO Γ

Composition typing rule

$$\frac{\Gamma \vdash \gamma : \Delta \qquad \Delta \vdash \delta : \Theta}{\Gamma \vdash \gamma \circ \delta : \Theta}$$

The middle context Δ must match!

Read: $\gamma \circ \delta$ takes us from Θ to Γ via Δ

Explicit Subs: Context Typing

Explicit Subs: Context Typing

$$\frac{}{\vdash \diamond \ \mathsf{ctx}}(\textit{Ctx-Empty})$$

Explicit Subs: Context Typing

Explicit Subs: Context Typing

$$\frac{}{\vdash \diamond\ \mathsf{ctx}}(\mathit{Ctx\text{-}Empty})$$

$$\frac{\Gamma \vdash \alpha\ \mathsf{Type}}{\vdash \Gamma, \alpha\ \mathsf{ctx}}(\mathit{Ctx\text{-}Extend})$$

Explicit Subs: Type Typing

Explicit Subs: Type Typing

$$\frac{\vdash \Gamma \text{ ctx}}{\Gamma \vdash * \text{ Type}}(\textit{Type-Base})$$

Explicit Subs: Type Typing

Explicit Subs: Type Typing

$$\frac{\vdash \Gamma \ \mathsf{ctx}}{\Gamma \vdash * \ \mathsf{Type}}(\textit{Type-Base})$$

$$\frac{\Gamma \vdash \alpha \ \mathsf{Type} \qquad \Gamma \vdash \beta \ \mathsf{Type}}{\Gamma \vdash \alpha \to \beta \ \mathsf{Type}}(\textit{Type-Arrow})$$

Explicit Subs: Term Typing

Explicit Subs: Term Typing

$$\frac{\vdash \Gamma, \alpha \ \mathsf{ctx}}{\Gamma, \alpha \vdash 1 : \alpha}(\mathit{Var})$$

Explicit Subs: Term Typing

Explicit Subs: Term Typing

$$\frac{\vdash \Gamma, \alpha \ \mathsf{ctx}}{\Gamma, \alpha \vdash 1 : \alpha}(\mathit{Var})$$

$$\frac{\Gamma, \alpha \vdash M : \beta}{\Gamma \vdash \lambda\alpha.M : \alpha \to \beta}(\mathit{Abs})$$

Explicit Subs: Term Typing

Explicit Subs: Term Typing

$$\frac{\vdash \Gamma, \alpha \ \mathsf{ctx}}{\Gamma, \alpha \vdash 1 : \alpha}(\mathit{Var})$$

$$\frac{\Gamma, \alpha \vdash M : \beta}{\Gamma \vdash \lambda\alpha.M : \alpha \to \beta}(\mathit{Abs})$$

$$\frac{\Gamma \vdash M : \alpha \to \beta \qquad \Gamma \vdash N : \alpha}{\Gamma \vdash M\ N : \beta}(\mathit{App})$$

Explicit Subs: Term Typing

Explicit Subs: Term Typing

$$\frac{\vdash \Gamma, \alpha \text{ ctx}}{\Gamma, \alpha \vdash 1 : \alpha}(\mathit{Var})$$

$$\frac{\Gamma, \alpha \vdash M : \beta}{\Gamma \vdash \lambda\alpha.M : \alpha \to \beta}(\mathit{Abs})$$

$$\frac{\Gamma \vdash M : \alpha \to \beta \qquad \Gamma \vdash N : \alpha}{\Gamma \vdash M\ N : \beta}(\mathit{App})$$

$$\frac{\Delta \vdash M : \alpha \qquad \Gamma \vdash \gamma : \Delta}{\Gamma \vdash M[\gamma] : \alpha}(\mathit{Subst})$$

Explicit Subs: Substitution Typing

Explicit Subs: Substitution Typing

$$\frac{\vdash \Gamma \ \mathsf{ctx}}{\Gamma \vdash \mathit{id} : \Gamma}(\mathit{Sub\text{-}Id})$$

Explicit Subs: Substitution Typing

Explicit Subs: Substitution Typing

$$\frac{\vdash \Gamma \text{ ctx}}{\Gamma \vdash \mathit{id} : \Gamma}(\mathit{Sub\text{-}Id})$$

$$\frac{\vdash \Gamma, \alpha \text{ ctx}}{\Gamma, \alpha \vdash \uparrow : \Gamma}(\mathit{Sub\text{-}Weak})$$

Explicit Subs: Substitution Typing

Explicit Subs: Substitution Typing

$$\frac{\vdash \Gamma \text{ ctx}}{\Gamma \vdash \mathit{id} : \Gamma}(\mathit{Sub\text{-}Id})$$

$$\frac{\vdash \Gamma, \alpha \text{ ctx}}{\Gamma, \alpha \vdash \uparrow : \Gamma}(\mathit{Sub\text{-}Weak})$$

$$\frac{\Gamma \vdash \gamma : \Delta \qquad \Gamma \vdash M : \alpha}{\Gamma \vdash \gamma, M : \alpha : \Delta, \alpha}(\mathit{Sub\text{-}Ext})$$

Explicit Subs: Substitution Typing

Explicit Subs: Substitution Typing

$$\frac{\vdash \Gamma \ \mathsf{ctx}}{\Gamma \vdash \mathit{id} \ : \Gamma}(\mathit{Sub\text{-}Id})$$

$$\frac{\vdash \Gamma, \alpha \ \mathsf{ctx}}{\Gamma, \alpha \vdash \uparrow \ : \Gamma}(\mathit{Sub\text{-}Weak})$$

$$\frac{\Gamma \vdash \gamma \ : \Delta \qquad \Gamma \vdash M \ : \alpha}{\Gamma \vdash \gamma, M : \alpha \ : \Delta, \alpha}(\mathit{Sub\text{-}Ext})$$

$$\frac{\Gamma \vdash \gamma \ : \Delta \qquad \Delta \vdash \delta \ : \Theta}{\Gamma \vdash \gamma \circ \delta \ : \Theta}(\mathit{Sub\text{-}Comp})$$

Equational Theory: Category Laws

Equational Theory: Category Laws

$$id \circ \gamma = \gamma \qquad\qquad \gamma \circ id = \gamma$$

Identity Laws

The identity substitution is both a left and right unit for composition

Equational Theory: Category Laws

Equational Theory: Category Laws

$$\mathit{id} \circ \gamma = \gamma \qquad\qquad \gamma \circ \mathit{id} = \gamma$$

$$(\gamma \circ \delta) \circ \theta = \gamma \circ (\delta \circ \theta)$$

Identity and Associativity Laws

Substitution composition forms a category

Equational Theory: Closure Laws

Equational Theory: Closure Laws

$$1[id] = 1$$

Variable Identity

Applying *id* to 1 is 1.

Equational Theory: Closure Laws

Equational Theory: Closure Laws

$$1[id] = 1$$

$$1[id, M : \alpha] = M$$

Variable Extension of Identity is Substitution.

Applying *id* extended with M to 1 is M. This is our actual substitution being performed.

Equational Theory: Closure Laws

$$1[id, M : \alpha] = M$$

$$1[\uparrow] = 1[\uparrow \circ \gamma]$$

$$M[\gamma][\delta] = M[\delta \circ \gamma]$$

Closure (Application) is composition

Note pay careful attention to the order of sequential closures here.

Equational Theory: Closure Laws

Equational Theory: Closure Laws

$$1[id, M : \alpha] = M$$

$$1[\uparrow] = 1[\uparrow \circ \gamma]$$

$$M[\gamma][\delta] = M[\delta \circ \gamma]$$

$$(\delta \circ \uparrow), 1[\delta] = \delta$$

Weaken, Extension

A substitution is equal to its most recent element extended on to the rest.

Equational Theory: Weakening Laws

Equational Theory: Weakening Laws

$$\uparrow, 1 = \mathit{id}$$

Weakening Extended with Var is Identity

Extending De-Bruijn index 0 with a new variable, and then removing it is the identity substitution.

Equational Theory: Weakening Laws

Equational Theory: Weakening Laws

$$\uparrow, 1 = \mathit{id}$$

$$\uparrow \circ \gamma, M : \alpha = \gamma$$

Weaken-Extend Cancel

Weakening an extended substituion recovers our original substitution.

Equational Theory: Distribution Laws

Equational Theory: Distribution Laws

$$(M\ N)[\gamma] = M[\gamma]\ N[\gamma]$$

Application Distribution

Substitution distributes over application.

Equational Theory: Distribution Laws

Equational Theory: Distribution Laws

$$(M\ N)[\gamma] = M[\gamma]\ N[\gamma]$$

$$(\lambda\alpha.M)[\gamma] = \lambda\alpha.M[(\uparrow \circ\gamma), 1 : \alpha]$$

Abstraction Distribution

Substitution distributes under binders by extending a weakening with variable.

Equational Theory: Extension Laws

Equational Theory: Extension Laws

$$\delta \circ (\gamma, M : \alpha) = (\delta \circ \gamma), M[\delta]$$

Extension Composition

Composition distributes into extensions.

Equational Theory: Extension Laws

Equational Theory: Extension Laws

$$(\gamma, M : \alpha) \circ \delta = (\gamma \circ \delta, M[\delta] : \alpha)$$

$$id = (\uparrow, 1 : \alpha)$$

Identity Decomposition

Identity as shift extended with variable.

Equational Theory: Computation Laws

Equational Theory: Computation Laws

$$(\lambda\alpha.M)\ N = M[id, N : \alpha]$$

Beta Reduction

Application reduces by substituting the argument

Equational Theory: Computation Laws

Equational Theory: Computation Laws

$$(\lambda\alpha.M)\ N = M[id, N : \alpha]$$

$$M = \lambda\alpha.(M[\uparrow]\ 1)$$

Eta Expansion

Any term can be eta-expanded by shifting and applying to variable 1

Equational Theory: Computation Laws

Equational Theory: Computation Laws

$$(\lambda\alpha.M)\ N = M[id, N : \alpha]$$

$$M = \lambda\alpha.(M[\uparrow]\ 1)$$

$$1[id, M : \alpha] = M$$

Variable Beta

Substituting for variable 1 with extension gives the substituted term

Equational Theory Intuition

Equational Theory Intuition

- Equations let us rewrite expressions to canonical forms
- Substitutions can be simplified before application
- No need for complex variable renaming (α-conversion)

Conclusions

Conclusion

We've seen how explicit substitutions:

- Make substitution a syntactic operation
- Use De Bruijn indices to avoid variable names
- Form a category with composition and identity
- Have a complete equational theory

Conclusions

Conclusion

We've seen how explicit substitutions:

- Make substitution a syntactic operation
- Use De Bruijn indices to avoid variable names
- Form a category with composition and identity
- Have a complete equational theory

Benefits

- Cleaner metatheory and proofs
- Direct implementation of theory
- Foundation for dependent type theory

Conclusions

Next Steps

This framework extends naturally to:

- Dependent type theory (as hinted)
- Proof assistants and type checkers
- Abstract machines for evaluation
- Categorical semantics of type theory

Conclusions

Thank You!

Questions?

Slides available at:
https://github.com/donovancrichton/Talks