Alpha-Structural Induction and Recursion for the Lambda Calculus in Constructive Type Theory

Ernesto Copello,Álvaro Tasistro,Nora Szasz, Ana Bove, Maribel Fernández

10th Workshop on Logical and Semantic Frameworks, with Applications.

Outline

Motivation

Reasoning over α-equivalence classes

Motivation

Studying and formalising reasoning techniques over programming languages.

- like pen-and-paper ones
- using constructive type theory as proof assistant

More specifically: λ-calculus & Agda.

Motivation

Studying and formalising reasoning techniques over programming languages.

- like pen-and-paper ones
- using constructive type theory as proof assistant

More specifically: λ-calculus & Agda.

Approach	Pros	Cons
One sort variables	Simple one	capture avoiding substitution (Church 36 / Curry-Feys 58) reasoning over α-eq. classes (*BVC*)

Motivation

Studying and formalising reasoning techniques over programming languages.

- like pen-and-paper ones
- using constructive type theory as proof assistant

More specifically: λ-calculus & Agda.

Approach	Pros	Cons
One sort variables	Simple one	capture avoiding substitution (Church 36 / Curry-Feys 58) reasoning over α-eq. classes (*BVC*)
Frege	substitution	reasoning over α-eq. classes

Motivation

Studying and formalising reasoning techniques over programming languages.

- like pen-and-paper ones
- using constructive type theory as proof assistant

More specifically: λ-calculus & Agda.

Approach	Pros	Cons
One sort variables	Simple one	capture avoiding substitution (Church 36 / Curry-Feys 58) reasoning over α-eq. classes (*BVC*)
Frege	substitution	reasoning over α-eq. classes
De Bruijn	α-eq. classes	indexes operations + well-formed predicates ——— distance from intuitive proofs

Motivation

Studying and formalising reasoning techniques over programming languages.

- like pen-and-paper ones
- using constructive type theory as proof assistant

More specifically: λ-calculus & Agda.

Approach	Pros	Cons
One sort variables	Simple one	capture avoiding substitution (Church 36 / Curry-Feys 58) reasoning over α-eq. classes (*BVC*)
Frege	substitution	reasoning over α-eq. classes
De Bruijn	α-eq. classes	indexes operations + well-formed predicates distance from intuitive proofs
Locally Nameless	less indexes operations	distance from intuitive proofs

Motivation

Studying and formalising reasoning techniques over programming languages.

- like pen-and-paper ones
- using constructive type theory as proof assistant

More specifically: λ-calculus & Agda.

Approach	Pros	Cons
One sort variables	Simple one	capture avoiding substitution (Church 36 / Curry-Feys 58) reasoning over α-eq. classes (*BVC*)
Frege	substitution	reasoning over α-eq. classes
De Bruijn	α-eq. classes	indexes operations + well-formed predicates distance from intuitive proofs
Locally Nameless	less indexes operations	distance from intuitive proofs
HOAS	substitution	has limitations

Motivation

Studying and formalising reasoning techniques over programming languages.

- like pen-and-paper ones
- using constructive type theory as proof assistant

More specifically: λ-calculus & Agda.

Approach	Pros	Cons
One sort variables	Simple one	capture avoiding substitution (Church 36 / Curry-Feys 58) reasoning over α-eq. classes (*BVC*)
Frege	substitution	reasoning over α-eq. classes
De Bruijn	α-eq. classes	indexes operations + well-formed predicates ——— distance from intuitive proofs
Locally Nameless	less indexes operations	distance from intuitive proofs
HOAS	substitution	has limitations
Nominal	swapping α-eq. classes	introduce swapping lemmas choice axiom incompatible

Reasoning over α-equivalence classes

Barenregt's variable convention

Each λ-term represents its α class, so it could be assumed to have bound and context variables all different.

Reasoning over α-equivalence classes

Barenregt's variable convention

Each λ-term represents its α class, so it could be assumed to have bound and context variables all different.

Formilising α-structural induction principle in a classic approach

- Complete induction over the *size* of terms is needed to fill the gap between terms and α-equivalence classes.
- Prove that the property being proved is α-equivalent.

Reasoning over α-equivalence classes

Barenregt's variable convention

Each λ-term represents its α class, so it could be assumed to have bound and context variables all different.

Formilising α-structural induction principle in a classic approach

- Complete induction over the *size* of terms is needed to fill the gap between terms and α-equivalence classes.
- Prove that the property being proved is α-equivalent.

Complete induction scketch

<u>λ case:</u> To prove $\forall x, M,\ P(\lambda x M)$ we can instead prove:

$$\exists x^*, M^* \text{ renamings}/\lambda x M \sim_\alpha \lambda x^* M^*,\ P(M^*) \Rightarrow P(\lambda x^* M^*)$$