The Barendregt-Geuvers-Klop Conjecture

Alejandro Pinilla Barrera
Juan Carlos Agudelo Agudelo

Universidad de Antioquia

June 2025

Table of contents

1 λ-calculus

2 Simply Typed λ-calculus

3 Pure Type Systems

4 The Barendregt-Geuvers-Klop Conjecture

5 References

Table of contents

1 λ-calculus

2 Simply Typed λ-calculus

3 Pure Type Systems

4 The Barendregt-Geuvers-Klop Conjecture

5 References

Introduction

The λ-calculus is a model of computation introduced a few years before another such model, Turing machines.

- With Turing machines, computation is expressed by reading from and writing to a tape, and performing actions depending on the context of the tape.
- In contrast in λ-calculus one is concerned with functions, and these may both take other functions as arguments and returns functions as results.

λ-terms

Definition 1: The set of λ-terms

Let $\mathbb{V} = \{x, y, z, \ldots\}$ be the set of infinite term variables. The **set of λ-terms** (Λ) is defined inductevely by:

- If $x \in \mathbb{V}$, then $x \in \Lambda$.
- If $M, N \in \Lambda$, then $MN \in \Lambda$.
- If $x \in \mathbb{V}$ and $M \in \Lambda$, then $\lambda x.M \in \Lambda$.

Remark 1: This definition can be summarized by the following grammar:

$$\Lambda ::= \mathbb{V} \mid \Lambda\Lambda \mid \lambda\mathbb{V}.\Lambda.$$

Remark 2: The functions in the λ-calculus have the property that the name of the variables are not essential. This can be formalized using α-conversion or the Barendregt convention.

Free variables

Definition 2: The set of free variables

The **set of free variables** of a λ-term can be inductively defined by:

1. $FV(x) = \{x\}$.
2. $FV(MN) = FV(M) \cup FV(N)$.
3. $FV(\lambda x.M) = FV(M) \setminus \{x\}$.

Example 1:

- $FV(\lambda x.xy) = \{y\}$.
- $FV(x(\lambda x.xy)) = \{x, y\}$.

Closed terms

Definition 3: Closed terms

A λ-term M is **closed** if $FV(M) = \emptyset$.

Remark 3: A closed term is also called a **combinator**.

Example 2:

- $\lambda xyz.xxy$ y $\lambda xy.xxy$ are closed λ-terms.
- $\lambda x.xxy$ is not a closed λ-term.

Substitution

Definition 4: Substitution

Let $N \in \Lambda$. The **substitution** operation is inductively defined by:

1. $x[x := N] = N$.
2. $y[x := N] = y$.
3. $PQ[x := N] = P[x := N]Q[x := N]$.
4. $(\lambda y.P)[x := N] = \lambda y.P[x := N]$ if $y \notin FV(N)$ and $y \neq x$.

β-reduction

Definition 5: Compatible Relation

A relation $\succ$ on Λ is **compatible** iff satisfies the following conditions for all $M, N, Z \in \Lambda$.

- If $M \succ N$, then $\lambda x.M \succ \lambda x.N$ for all x.
- If $M \succ N$, then $MZ \succ NZ$.
- If $M \succ N$, then $ZM \succ ZN$.

β-reduction

Definition 6: β-reduction

The least compatible relation $\rightarrow_\beta$ on Λ satisfying:

$$(\lambda x.P)Q \rightarrow_\beta P[x := Q],$$

is called β-**reduction**.

Remark 4:

- A term of the form $(\lambda x.P)Q$ is called a β-**redex**.
- The term $P[x := Q]$ is said to arise by contracting the redex.

β-reduction

Definition 7: Multi-step β-reduction

The **multi-step β-reduction** ($\twoheadrightarrow_\beta$) is the transitive and reflexive closure of $\rightarrow_\beta$.

Remark 5:

- The relation β-**equality** or β-**conversion** ($=_\beta$) is the least equivalence relation containing $\rightarrow_\beta$.
- A β-**reduction** sequence is a finite or infinite sequence $M_0 \rightarrow_\beta M_1 \rightarrow_\beta M_2 \rightarrow_\beta \cdots$.

Example 3:

- $(\lambda x.xx)(\lambda z.z) \twoheadrightarrow_\beta \lambda y.y$.
- $(\lambda x.x)yz =_\beta y((\lambda x.x)z)$.

β-normal forms

Definition 8: β-normal form

A λ-term is in β-**normal** form ($M \in \mathrm{NF}_\beta$) iff there is no N such that $M \to_\beta N$, i. e. M does not contain a β-redex.

Example 4:

- $\lambda x.x$ is in β-normal form.
- x is in β-normal form.

Remark 6: A term is in normal form iff it is an abstraction $\lambda x.M$, where M is in normal form, or it is $xM_1M_2\cdots M_n$, where $n \geq 0$ and all M_i are in normal form. Even more compact: a normal form is $\lambda y_1 \ldots y_m.xM_1M_2\cdots M_n$, where $m, n \geq 0$ and all M_i are normal forms.

β-normal forms

Lemma 1

1. If $N \to_\beta N'$, then $M[x := N] \twoheadrightarrow_\beta M[x := N']$.
2. If $M \to_\beta M'$ then $M[x := N] \to_\beta M'[x := N]$.

Proof.

By induction on M and $M \to_\beta M'$ using properties of substitution. □

Normalizing and Strongly Normalizing

Definition 9: Normalizing and Strongly Normalizing

1. A term M is **normalizing** ($N \in \mathrm{WN}_\beta$) iff there is a reduction sequence from M ending in a normal form N.
2. A term M is **strongly normalizing** ($M \in \mathrm{SN}_\beta$) if all reductions sequences starting with M are finite.

Remark 7:

- If $M \in \mathrm{WN}_\beta$, we then say that M has the normal form N.
- We write $M \in \infty_\beta$ if $M \notin \mathrm{SN}_\beta$.
- Any strongly normalizing term is also normalizing but the converse is not true, as $(\lambda xy.y)((\lambda w.ww)(\lambda w.ww))$ shows.

Leftmost reductions are normalizing

Notation 1: Let $n \geq 0$. If $\overrightarrow{P} = P_1, \ldots, P_n$, then we write $M\overrightarrow{P}$ for $MP_1 \cdots P_n$. Similarly, if $\overrightarrow{z} = z_1, \ldots, z_n$, then we write $\lambda\overrightarrow{z}.M$ for $\lambda z_1 \cdots z_n.M$.

Remark 8: Any term has exactly one of the following forms $\lambda\overrightarrow{z}.x\overrightarrow{R}$ or $\lambda\overrightarrow{z}.(\lambda x.P)Q\overrightarrow{R}$, in which case $(\lambda x.P)Q$ is called **head redex**. Any redex that is not a head redex is called **internal**. A head redex is always the leftmost redex, but the leftmost redex in a term is not necessarily a head redex.

Definition 10

For a term M not in normal form, we write

- $M \overset{l}{\rightarrow}_\beta N$ if N arises from M by contracting the leftmost redex.
- $M \overset{h}{\rightarrow}_\beta N$ if N arises from M by contracting a head redex.
- $M \overset{i}{\rightarrow}_\beta N$ if N arises from M by contracting an internal redex.

Leftmost reductions are normalizing

Lemma 2

1. If $M \xrightarrow{h}_{\beta} N$, then $\lambda x.M \xrightarrow{h}_{\beta} \lambda x.N$.
2. If $M \xrightarrow{h}_{\beta} N$ and M is not an abstraction, then $ML \xrightarrow{h}_{\beta} NL$.
3. If $M \xrightarrow{h}_{\beta} N$, then $M[x := L] \xrightarrow{h}_{\beta} N[x := L]$.

Proof.

Direct. □

Leftmost reductions are normalizing

Definition 11: Parallel Reductions

Let $\Rightarrow_\beta$ be the least relation on Λ such that:

- $x \Rightarrow_\beta x$ for all variables x.
- If $P \Rightarrow_\beta Q$, then $\lambda x.P \Rightarrow_\beta \lambda x.Q$.
- If $P_1 \Rightarrow_\beta Q_1$ and $P_2 \Rightarrow_\beta Q_2$, then $P_1 P_2 \Rightarrow_\beta Q_1 Q_2$.
- If $P_1 \Rightarrow_\beta Q_1$ and $P_2 \Rightarrow_\beta Q_2$, then $(\lambda x.P_1)P_2 \Rightarrow_\beta Q_1[x := Q_2]$.

Leftmost reductions are normalizing

Lemma 3

1. If $M \rightarrow_\beta N$, then $M \Rightarrow_\beta N$.
2. If $M \Rightarrow_\beta N$, then $M \twoheadrightarrow_\beta N$.
3. If $M \Rightarrow_\beta M'$ and $N \Rightarrow_\beta N'$, then $M[x := N] \Rightarrow_\beta M'[x := N']$.

Proof.

1. is by induction on the definition of $M \rightarrow_\beta N$, and 2., 3. are by induction on the definition of $M \Rightarrow_\beta M'$. □

Leftmost reductions are normalizing

Remark 9: We write $\overrightarrow{P} \Rightarrow_\beta \overrightarrow{Q}$ if $\overrightarrow{P} = P_1, \ldots, P_n$, $\overrightarrow{Q} = Q_1, \ldots, Q_n$, $n \geq 0$, and $P_j \Rightarrow_\beta Q_j$ for all $1 \leq j \leq n$.

Definition 12: Parallel internal reduction

Parallel internal reduction $\overset{i}{\Rightarrow}_\beta$ is the least relation on Λ satisfying the following rules:

- $\overrightarrow{P} \Rightarrow_\beta \overrightarrow{Q}$, then $\lambda \overrightarrow{x}.y\overrightarrow{P} \overset{i}{\Rightarrow}_\beta \lambda \overrightarrow{x}.y\overrightarrow{Q}$.
- If $\overrightarrow{P} \Rightarrow_\beta \overrightarrow{Q}$, $S \Rightarrow_\beta T$ and $R \Rightarrow_\beta U$, then $\lambda \overrightarrow{x}.(\lambda y.S)R\overrightarrow{P} \overset{i}{\Rightarrow}_\beta \lambda \overrightarrow{x}.(\lambda y.T)U\overrightarrow{Q}$.

Remark 10: If $M \overset{i}{\rightarrow}_\beta N$, then $M \overset{i}{\Rightarrow}_\beta N$. Conversely, if $M \overset{i}{\Rightarrow}_\beta N$, then $M \overset{i}{\twoheadrightarrow}_\beta N$. Also, if $M \overset{i}{\Rightarrow}_\beta N$, then $M \Rightarrow_\beta N$.

Leftmost reductions are normalizing

Definition 13

We write $M \Rrightarrow N$ if there are $M_0, M_1, \dots, M_n$ with

$$M = M_0 \overset{h}{\rightarrow}_{\beta} M_1 \overset{h}{\rightarrow}_{\beta} \cdots \overset{h}{\rightarrow}_{\beta} M_n \overset{i}{\Rightarrow}_{\beta} N$$

and $M_i \Rightarrow_{\beta} N$ for all $i \in \{0, 1, \dots, n\}$, where $n \geq 0$.

Leftmost reductions are normalizing

Lemma 4

1. If $M \Rrightarrow_\beta M'$, then $\lambda x.M \Rrightarrow_\beta \lambda x.M'$.
2. If $M \Rrightarrow_\beta M'$ and $N \Rrightarrow_\beta N'$, then $MN \Rrightarrow_\beta M'N'$.
3. If $M \Rrightarrow_\beta M'$ and $N \Rrightarrow_\beta N'$, then $M[x := N] \Rrightarrow_\beta M'[x := N']$.

Proof.

See Sørensen and Urzyczyn (2006). □

Leftmost reductions are normalizing

Lemma 5

1. If $M \Rightarrow_\beta N$, then $M \overset{h}{\twoheadrightarrow}_\beta L \overset{i}{\Rightarrow}_\beta N$ for some L.
2. If $M \overset{i}{\Rightarrow}_\beta N \overset{h}{\rightarrow}_\beta L$, then $M \overset{h}{\twoheadrightarrow}_\beta O \overset{i}{\Rightarrow}_\beta L$ for some O.

Proof.

See Sørensen and Urzyczyn (2006). □

Leftmost reductions are normalizing

Theorem 1

If M has a normal form, then $M \overset{l}{\twoheadrightarrow}_\beta N$.

Proof.

Induction on the length of N and the previous Lemma. $\square$

Leftmost reductions are normalizing

Definition 14: Reduction Strategy

A **reduction strategy** F is a map from λ-terms to λ-terms such that $F(M) = M$ when M is in normal form, and $M \to_\beta F(M)$ otherwise. Such and F is **normalizing** if for all $M \in \mathrm{WN}_\beta$, there is an i such that $F^i(M)$ is in normal form.

Corollary 1

Define $F_l(M) = M$ for each normal form M, and $F_l(M) = N$, where $M \xrightarrow{l}_\beta N$, otherwise. Then F_l is normalizing.

Leftmost reductions are normalizing

Definition 15

A reduction sequence is called

- **quasi-leftmost** if it contains infinitely many leftmost reductions.
- **quasi-head** if it contains infinitely many head reductions.

Corollary 2

Let M be normalizing. Then

1. M has no infinite head-reduction sequence.
2. M has no quasi-head reduction sequence.
3. M has no quasi-leftmost reduction sequence.

Perpetual reductions

Definition 16

Define $F_\infty : \Lambda \longrightarrow \Lambda$ as follows. If $M \in \mathrm{NF}_\beta$, then $F_\infty(M) = M$, otherwise,

- $F_\infty(\lambda\vec{z}.x\vec{P}Q\vec{R}) = \lambda\vec{z}.x\vec{P}F_\infty(Q)\vec{R}$ if $\vec{P} \in \mathrm{NF}_\beta$ and $Q \notin \mathrm{NF}_\beta$.
- $F_\infty(\lambda\vec{z}.(\lambda x.P)Q\vec{R}) = \lambda\vec{z}.P[x := Q]\vec{R}$ if $x \in FV(P)$ or $Q \in \mathrm{NF}_\beta$.
- $F_\infty(\lambda\vec{z}.(\lambda x.P)Q\vec{R}) = \lambda\vec{z}.(\lambda x.P)F_\infty(Q)\vec{R}$ if $x \notin FV(P)$ and $Q \notin \mathrm{NF}_\beta$.

Remark 11: $M \to_\beta F_\infty(M)$ when $M \notin \mathrm{NF}_\beta$.

Perpetual reductions

Lemma 6

Assume $Q \in \mathrm{SN}_\beta$ or $x \in FV(P)$. If $P[x := Q]\vec{R} \in \mathrm{SN}_\beta$, then $(\lambda x.P)Q\vec{R} \in \mathrm{SN}_\beta$.

Proof.

See Sørensen and Urzyczyn (2006). □

Theorem 2

If $M \in \infty_\beta$, then $F_\infty(M) \in \infty_\beta$.

Proof.

By induction on M and considering the cases in the definition of $F_\infty(\cdot)$. □

Perpetual reductions

Definition 17: Perpetual reduction strategy

A reduction strategy F is perpetual iff for all $M \in \infty_\beta$,

$$M \to_\beta F(M) \to_\beta F(F(M)) \to_\beta \cdots$$

is an infinite reduction sequence starting from M.

Corollary 3

F_∞ is perpetual.

Conservation theorem

Definition 18: λI-terms

The set of λI-terms is defined as follows:

- Every variable is a λI-term.
- An application MN is a λI-term iff both M and N are λI-terms.
- An abstraction $\lambda x.M$ is a λI-term iff M is a λI-term and $x \in FV(M)$.

Example 5:

- $\lambda x.x$ is a λI-term.
- $\lambda x.y$ is not a λI-term.

Conservation theorem

Theorem 3: The conservation theorem

1. For all λI-terms M, if $M \in \mathrm{WN}_\beta$, then $M \in \mathrm{SN}_\beta$.
2. For all λI-terms M, if $M \in \infty_\beta$ and $M \to_\beta N$, then $N \in \infty_\beta$.

Proof.

For part 1. assume $M \in \mathrm{WN}_\beta$. Then by Theorem 1 $M \overset{l}{\twoheadrightarrow}_\beta N$ for some normal form N. Now, note that for all λI-terms L not in normal form, $L \overset{l}{\to}_\beta F_\infty(L)$. Thus $N = F_\infty^k(M)$ for some k, so $M \in \mathrm{SN}_\beta$ by Corollary 1. For part 2., assume $M \to_\beta N$. If $M \in \infty_\beta$, then $M \notin \mathrm{SN}_\beta$, then $M \notin \mathrm{WN}_\beta$, by 1. Hence $N \notin \mathrm{WN}_\beta$, in particular $N \in \infty_\beta$. □

Table of contents

1 λ-calculus

2 Simply Typed λ-calculus

3 Pure Type Systems

4 The Barendregt-Geuvers-Klop Conjecture

5 References

Introduction

As we have seen in the first part, the abstract behavior of functions can be expressed very well by means of λ-calculus, however, as Nederpelt and Geuvers (2014) point out: λ-calculus is sometimes too 'liberal' to conform to our intuitive demands concerning functions and how they should act as input-output devices.

Functions are usually thought of as acting on elements belonging to a certain collection or domain. This notion can be carried out in the λ-calculus by introducing **types**. These new objects are intended to give certain restrictions on the input values permitted.

The set of types

Definition 19: The set $\mathbb{T}$ of types

Let $\mathbb{B} = \{\alpha, \beta, \gamma, \dots\}$ be the **set of basic types**. The **set of types** $\mathbb{T}$ is defined by:

- If $\alpha \in \mathbb{B}$, then $\alpha \in \mathbb{T}$.
- If $\sigma, \tau \in \mathbb{T}$, then $\sigma \to \tau \in \mathbb{T}$.

Remark 12: This definition can be summarized by the following grammar:

$$\mathbb{T} ::= \mathbb{B} \mid \mathbb{T} \to \mathbb{T}.$$

Example 6:

- $\beta \to \gamma$.
- $((\gamma \to \alpha) \to (\alpha \to (\beta \to \gamma)))$.

Judgement

Definition 20

- A **statement** is of the form $M : \sigma$, where $M \in \Lambda$ and $\sigma \in \mathbb{T}$. In such statement, M is called the subject and σ the type.
- A **declaration** is a statement with a variable as a subject.
- A **context** is a list of declarations with different subjects.
- A **judgement** has the form $\Gamma \vdash M : \sigma$, with Γ a context and $M : \sigma$ a statement.

Derivation rules

Definition 21: Derivation rules

We say that a judgement $\Gamma \vdash M : \sigma$ is valid if it can be generated by the following derivation rules:

$$\boxed{\begin{array}{l} \text{Var: } \Gamma, x : \sigma \vdash x : \sigma \\[1em] \text{Abs: } \dfrac{\Gamma, x : \sigma \vdash M : \tau}{\Gamma \vdash \lambda x : \sigma . M : \sigma \to \tau} \\[1em] \text{App: } \dfrac{\Gamma \vdash M : \sigma \to \tau \qquad \Gamma \vdash N : \sigma}{\Gamma \vdash MN : \tau} \end{array}}$$

β-reduction

Definition 22: β-reduction

The relation $\to_\beta$ is the least compatible relation such that

$$(\lambda x : \sigma.P)Q : \tau \to_\beta P[x := Q] : \tau.$$

Theorem 4: The subject reduction theorem

If $\Gamma \vdash M : \sigma$ and $M : \sigma \twoheadrightarrow_\beta N : \sigma$, then $\Gamma \vdash N : \sigma$.

Proof.

See Sørensen and Urzyczyn (2006). $\square$

Normalization

Theorem 5: The weak normalization theorem

Every term of the simply typed λ-calculus has a normal form.

Theorem 6: The strong normalization theorem

Every term of the simply typed λ-calculus is strongly normalizing.

Proof.

The idea of the proof is to infer the strong property from the weak one. This can be done with the help of The conservation theorem for the λI-terms by translating an arbitrary typed λ-term M into a λI-term $\iota(M)$ of the same type, such that $\iota(M) \in \mathsf{SN}_\beta$ implies that $M \in \mathsf{SN}_\beta$ (see Sørensen & Urzyczyn, 2006, for the details of the proof). $\square$

Table of contents

1. λ-calculus
2. Simply Typed λ-calculus
3. **Pure Type Systems**
4. The Barendregt-Geuvers-Klop Conjecture
5. References

Introduction

From the simply typed λ-calculus, more general type systems have been introduced to increase the expressiveness of the λ-calculus allowing many more applications in logic or computer science.

An example of this is the λ-cube (Barendregt, 1991), a framework to investigate the different dimensions in which the calculus of constructions (Coquand & Huet, 1986) is a generalization of the simply typed λ-calculus.

The λ-cube

$$\begin{tikzcd}
 & \lambda\omega \arrow[rr] & & \lambda C \\
\lambda 2 \arrow[ur] \arrow[rr] & & \lambda P2 \arrow[ur] & \\
 & \lambda\,\underline{\omega} \arrow[uu, dashed] \arrow[rr, dashed] & & \lambda P\underline{\omega} \arrow[uu] \\
\lambda_{\to} \arrow[uu] \arrow[ur, dashed] \arrow[rr] & & \lambda P \arrow[uu] \arrow[ur] &
\end{tikzcd}$$

Figure 1: Lambda cube. Direction of each arrow is direction of inclusion.

The λ-cube

Each dimension of the λ-cube corresponds to a kind of dependency[1] between terms and types:

- x-axis ($\rightarrow$): types that can depend on terms.
- y-axis ($\uparrow$): terms that can depend on types.
- z-axis ($\nearrow$): types that can depend on other types.

The different ways to combine these three dimensions yield the 8 vertices of the cube, each one corresponding to a different type system.

[1] Here dependency means the capacity of a term or type to bind a term or type.

The λ-cube

- $\lambda_{\rightarrow}$ is the simply typed λ-calculus (Church, 1940).
- $\lambda 2$ is the polymorphic system or second order typed λ-calculus and is a subsystem of the system F (Girard, 1972).
- $\lambda\omega$ is essentially the system $F\omega$ of Girard (1972).
- λP corresponds to one the systems in the family AUTOMATH languages (De Bruijn, 1980). It also appears under the name LF in Harper, Honsell, and Plotkin (1993).

The λ-cube

- $\lambda P2$ is studied in Longo and Moggi (1988).
- λC is one of the versions of the theory of constructions introduced by Coquand and Huet (1986).
- $\lambda\underline{\omega}$ is related to the POLYREC system studied by De Lavalette (1985).
- $\lambda P\underline{\omega}$ seems not to have been studied before.

Pure type systems

The method of generating the systems in the λ-cube has been generalized independently by Berardi (1990) and Terlouw (1989) which resulted in the notion of generalized type systems (GTS) or pure type systems (PTS).

The success of the PTS is concerned with logic thanks to the result of the so called **propositions-as-types** interpretation.

Definition 23: Pure type systems

A **PTS** is a triple $(\mathcal{S}, \mathcal{A}, \mathcal{R})$ where:

1. $\mathcal{S}$ is a set of **sorts**.
2. $\mathcal{A} \subseteq \mathcal{S} \times \mathcal{S}$ is a set of **axioms**.
3. $\mathcal{R} \subseteq \mathcal{S} \times \mathcal{S} \times \mathcal{S}$ is a set of **rules**.

Pure type systems

Definition 24: The set of variables

For each $s \in \mathcal{S}$, let $\mathbb{V}_s$ denote the countable infinite **set of variables** such that $\mathbb{V}_s \cap \mathbb{V}_{s'} = \emptyset$ when $s \neq s'$ and let $\mathbb{V} = \bigcup\limits_{s \in \mathcal{S}} \mathbb{V}_s$.

Definition 25: The set of terms

The **set $\mathcal{T}$ of terms** is given by the following syntax:

$$\mathcal{T} ::= \mathbb{V} | \mathcal{S} | \mathcal{T}\mathcal{T} | \lambda \mathbb{V} : \mathcal{T}.\mathcal{T} | \Pi \mathbb{V} : \mathcal{T}.\mathcal{T}.$$

Pure type systems

Definition 26: β-reduction

The least compatible relation $\rightarrow_\beta$ on $\mathcal{T}$ satisfying:

$$(\lambda x : A.M)N \rightarrow_\beta M[x := N]$$

is called β-**reduction**.

Remark 13: The concepts of **statement**, **declaration**, **context** and **judgement** are defined as for the $\lambda_\rightarrow$.

Pure type systems

A PTS, denoted by $\lambda\mathcal{S}$, is determined by the specification of $(\mathcal{S}, \mathcal{A}, \mathcal{R})$. The derivation rules for PTS are defined by the following axioms and rules:

Axiom: $\vdash s_1 : s_2$ if $(s_1, s_2) \in \mathcal{A}$.

Start: $\dfrac{\Gamma \vdash A : s}{\Gamma, x : A \vdash x : A}$ if $x \in \mathbb{V}_s$ and $x \notin \mathrm{dom}(\Gamma)$.

Weakening: $\dfrac{\Gamma \vdash B : C \qquad \Gamma \vdash A : s}{\Gamma, x : A \vdash B : C}$ if $x \in \mathbb{V}_s$ and $x \notin \mathrm{dom}(\Gamma)$.

Product: $\dfrac{\Gamma \vdash A : s_1 \qquad \Gamma, x : A \vdash B : s_2}{\Gamma \vdash (\Pi x : A.B) : s_3}$ if $(s_1, s_2, s_3) \in \mathcal{R}$.

Pure type systems

Application: $\dfrac{\Gamma \vdash F : (\Pi x : A.B) \qquad \Gamma \vdash a : A}{\Gamma \vdash Fa : B[x := a]}$

Abstraction: $\dfrac{\Gamma, x : A \vdash b : B \qquad \Gamma \vdash (\Pi x : A.B) : s}{\Gamma \vdash \lambda x : A.b : \Pi x : A.B}$.

Conversion: $\dfrac{\Gamma \vdash A : B \qquad \Gamma \vdash B' : s}{\Gamma \vdash A : B'}$ if $B =_\beta B'$.

Pure type systems

Example 7: The system $\lambda *$ is defined by:

1. $\mathcal{S} = \{*\}$.
2. $\mathcal{A} = \{(*,*)\}$.
3. $\mathcal{R} = \{(*,*)\}$.

Example 8: The λ-cube consists of eight PTS λS where:

1. $\mathcal{S} = \{*, \Box\}$.
2. $\mathcal{A} = \{(*, \Box)\}$.
3. $\{(*,*)\} \subseteq \mathcal{R} \subseteq \{(*,*), (\Box,*), (*,\Box), (\Box,\Box)\}$.

The name of each system and its associated set of rules is given in the following table.

Pure type systems

$\lambda_{\rightarrow}$	$(*, *)$			
$\lambda 2$	$(*, *)$	$(\square, *)$		
$\lambda \underline{\omega}$	$(*, *)$		$(\square, \square)$	
$\lambda \omega$	$(*, *)$	$(\square, *)$	$(\square, \square)$	
λP	$(*, *)$			$(*, \square)$
$\lambda P2$	$(*, *)$	$(\square, *)$		$(*, \square)$
$\lambda P \underline{\omega}$	$(*, *)$		$(\square, \square)$	$(*, \square)$
λC	$(*, *)$	$(\square, *)$	$(\square, \square)$	$(*, \square)$

Table 1: The systems of the λ-cube with their corresponding set of rules.

Pure type systems

Example 9:

- The system λHOL is defined by:
  1. $\mathcal{S} = \{*, \Box, \triangle\}$.
  2. $\mathcal{A} = \{(*, \Box), (\Box, \triangle)\}$.
  3. $\mathcal{R} = \{(*, *), (\Box, *), (\Box, \Box)\}$.
- The system λU^- is defined by:
  1. $\mathcal{S} = \{*, \Box, \triangle\}$.
  2. $\mathcal{A} = \{(*, \Box), (\Box, \triangle)\}$.
  3. $\mathcal{R} = \{(*, *), (\Box, *), (\Box, \Box), (\triangle, \Box)\}$.
- The system λU is defined by:
  1. $\mathcal{S} = \{*, \Box, \triangle\}$.
  2. $\mathcal{A} = \{(*, \Box), (\Box, \triangle)\}$.
  3. $\mathcal{R} = \{(*, *), (\Box, *), (\Box, \Box), (\triangle, *), (\triangle, \Box)\}$.

Table of contents

1 λ-calculus

2 Simply Typed λ-calculus

3 Pure Type Systems

4 **The Barendregt-Geuvers-Klop Conjecture**

5 References

The Barendregt-Geuvers-Klop conjecture

This conjecture was presented by Barendregt at *Type Lambda-Calculi and Applications* 1995.

The Barendregt-Geuvers-Klop conjecture

For every PTS, weak normalization implies strong normalization.

Remark 14: This is motivated primarily by a collection of results that derive strong normalization from weak normalization from systems in the λ-cube.

Example 10:

- All the systems of the λ-cube are strongly normalizing.
- The system $\lambda *$ is the simplest PTS which is not normalizing.

The Barendregt-Geuvers-Klop Conjecture

There are two natural techniques based on type-preserving translations for proving strong normalization from weak normalization:

1. The first one is to define a translation from expressions to λI-expressions. Strong normalization then readily follows from weak normalization by The Conservation Theorem for the untyped λ-calculus (Sørensen, 1997; Xi, 1997).
2. The second technique is to define an infinite-reduction-path-preserving translation to a weak system for which the conjecture is known to hold. Then the conjecture can be shown to hold for the full system (Geuvers & Nederhof, 1991; Harper & Lillibridge, 1993).

The Barendregt-Geuvers-Klop Conjecture

Some advances in the conjecture:

- Barthe, Hatcliff, and Sørensen (2001) extended the ideas of Sørensen (1997) to a class of PTS: the Generalized Non-Dependent PTS in which types do not depend on terms and includes $\lambda_{\rightarrow}$, $\lambda 2$, $\lambda \underline{\omega}$, $\lambda \omega$, λHOL, λU and λU^{-}. In this class, the Barendregt-Geuvers-Klop conjecture holds.
- Harper and Lillibridge (1993) showed that some extensions of arbitrary PTS with additional sorts and rules, which preserve the normalization property, can be used to provide results about the conjecture of some PTS.
- Mull (2022) showed that by introducing a new class of PTS: Tiered PTS the questions about normalization can be tackle easier. In fact, Mull presented a simpler and more approachable proof of the main result of Barthe et al. (2001).

The Barendregt-Geuvers-Klop Conjecture

Idea: Use the concept of λI-terms to proof the conjecture for all PTS or a class of them, taking advantage of The conservation theorem of untyped λ-calculus. To do this we can take two possible approaches:

1. Define translations from expressions to λI expressions extending the previous results of Sørensen (1997) and Barthe et al. (2001).
2. Extend the definition of λI-expressions, as it was done by Cartagena-Cartagena (2023) for the λ-cube, and generalize The conservation theorem.

Thank you!

Table of contents

1 λ-calculus

2 Simply Typed λ-calculus

3 Pure Type Systems

4 The Barendregt-Geuvers-Klop Conjecture

5 References

References I

Barendregt, H. P. (1991). Introduction to generalized type systems.

Barthe, G., Hatcliff, J., & Sørensen, M. H. (2001). Weak normalization implies strong normalization in a class of non-dependent pure type systems. *Theoretical Computer Science*, *269*(1-2), 317–361.

Berardi, S. (1990). *Type dependence and constructive mathematics* (Unpublished doctoral dissertation). PhD thesis, Dipartimento di Informatica, Torino, Italy.

Cartagena-Cartagena, D. (2023). *Relevance pure type systems* (Unpublished doctoral dissertation). Facultad de Ciencias Exactas y Naturales, Universidad de Antioquia, Colombia.

Church, A. (1940). A formulation of the simple theory of types. *The journal of symbolic logic*, *5*(2), 56–68.

Coquand, T., & Huet, G. (1986). *The calculus of constructions* (Unpublished doctoral dissertation). INRIA.

References II

De Bruijn, N. (1980). A survey of the project automath. In J. Seldin & J. Hindley (Eds.), *To H.B. Curry : Essays on combinatory logic, lambda calculus and formalism* (pp. 579–606). United States: Academic Press Inc.

De Lavalette, G. R. R. (1985). *Theories with type-free application and extended bar induction* (Unpublished doctoral dissertation). uitgever niet vastgesteld.

Geuvers, H., & Nederhof, M.-J. (1991). Modular proof of strong normalization for the calculus of constructions. *J. Funct. Program.*, *1*(2), 155–189.

Girard, J.-Y. (1972). *Interprétation fonctionnelle et élimination des coupures de l'arithmétique d'ordre supérieur* (Unpublished doctoral dissertation). Éditeur inconnu.

Harper, R., Honsell, F., & Plotkin, G. (1993). A framework for defining logics. *Journal of the ACM (JACM)*, *40*(1), 143–184.

References III

Harper, R., & Lillibridge, M. (1993). Polymorphic type assignment and cps conversion. *LISP and Symbolic Computation*, *6*(3), 361–379.

Longo, G., & Moggi, E. (1988). Constructive natural deduction and its ω-set interpretation. *Modest'Interpretation," Report CMU-CS-88-131, Computer Science Department, Carnegie Mellon University*.

Mull, N. (2022). *Strong normalization from weak normalization in non-dependent pure type systems via thunkification* (Tech. Rep.). Research Report, https://nmmull. github. io/thunk. pdf.

Nederpelt, R., & Geuvers, H. (2014). *Type theory and formal proof: an introduction*. Cambridge University Press.

Sørensen, M. H. (1997). Strong normalization from weak normalization in typedλ-calculi. *Information and Computation*, *133*(1), 35–71.

Sørensen, M. H., & Urzyczyn, P. (2006). *Lectures on the curry-howard isomorphism* (Vol. 149). Elsevier.

References IV

Terlouw, J. (1989). Een nadere bewijstheoretische analyse van gstt's. *Manuscript (in Dutch)*.

Xi, H. (1997). Weak and strong beta normalisations in typed λ-calculi. In *International conference on typed lambda calculi and applications* (pp. 390–404).