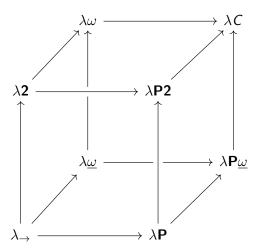
A general Subject Reduction for Pure Type Systems COMP527 — Logic and Computation

Charlotte Marchal, Dashiell Rich, Milton Rosenbaum, Zhaoshen Zhai

March 30, 2025

The Cube



Lots of Systems

In this diagram, eight systems are defined. The arrow represents \subseteq , so the weakest system is the simply typed lambda calculus, at the bottom left. Other named systems are $\lambda 2$, which is equivalent to system \mathbf{F} , which is the polymorphic lambda calculus. $\lambda \omega$ is equivalent to $F\omega$, which was proposed by Girard, λP is equivalent to the "automath" language, and is sometimes called LF.

Kinds of Dependencies

 $\begin{array}{c|c} (\star,\star) & \text{Terms can depend on terms} \\ (\square,\star) & \text{Terms can depend on types} \\ (\star,\square) & \text{Types can depend on terms} \\ (\square,\square) & \text{Types can depend on types} \\ \end{array}$

Parameterized & Homogeneous

Just change two two rules!

$$\Pi-rule \qquad \qquad \frac{\Gamma \vdash A: s_1 \quad \Gamma, x: A \vdash B: s_2}{\Gamma \vdash (\Pi x: A.B): s_3}$$

$$\lambda-rule \quad \frac{\Gamma\vdash A:s_1\quad \Gamma,x:A\vdash b:B\quad \Gamma,x:A\vdash B:s_2}{\Gamma(\lambda x:A.b):(\Pi x:A.B)}$$

In addition to \square and \star , we can add arbitrary sorts and axiom relations. A *pure type system* $\lambda(S,A,R)$ can be described as

ullet S : a set of sorts, for example \star,\Box

In addition to \square and \star , we can add arbitrary sorts and axiom relations. A *pure type system* $\lambda(S,A,R)$ can be described as

- ullet S : a set of sorts, for example \star , \Box
- A: a set of axioms, ★: □

In addition to \square and \star , we can add arbitrary sorts and axiom relations. A *pure type system* $\lambda(S, A, R)$ can be described as

- ullet S : a set of sorts, for example \star,\Box
- A: a set of axioms, ★: □
- R : a set of rules, the valid ways to parameterize the λ and Π rules. λC has all of the pairs of sorts.

In addition to \square and \star , we can add arbitrary sorts and axiom relations. A *pure type system* $\lambda(S, A, R)$ can be described as

- ullet S : a set of sorts, for example \star,\Box
- A: a set of axioms, ★: □
- R : a set of rules, the valid ways to parameterize the λ and Π rules. λC has all of the pairs of sorts.

We abbreviate 'pure type system' by PTS.

Even more systems

A surprising number of systems can be defined as PTSs:

•
$$\lambda C' = (\star^t, \star^p, \Box), (\star^t : \Box, \star^p : \Box), (S^2, \text{ that is, all pairs})$$

Even more systems

A surprising number of systems can be defined as PTSs:

- $\lambda C' = (\star^t, \star^p, \square), (\star^t : \square, \star^p : \square), (S^2, \text{ that is, all pairs})$
- $\bullet \ \lambda U = [\star, \square, \Delta], [\star : \square, \square : \Delta], [(\star, \star), (\square, \star), (\square, \square), (\Delta, \square), (\Delta, \star)]$

Mechanize Proof for Each PTS

Since so many systems are easily describable as PTSs, it would be nice to prove some properties about the systems in general.

In our project, we plan to:

• Prove Subject Reduction for arbitrary PTSs.

Mechanize Proof for Each PTS

Since so many systems are easily describable as PTSs, it would be nice to prove some properties about the systems in general.

In our project, we plan to:

- Prove Subject Reduction for arbitrary PTSs.
- Mechanize arbitrary PTSs in Beluga.

Tasks

• Milton: Implementing PTSs in Beluga and mechanizing the proof of Subject Reduction.

Tasks

- Milton: Implementing PTSs in Beluga and mechanizing the proof of Subject Reduction.
- Everyone else: Prove Subject Reduction for PTSs.

Tasks

- Milton: Implementing PTSs in Beluga and mechanizing the proof of Subject Reduction.
- Everyone else: Prove Subject Reduction for PTSs.
- Zhaoshen: Write up the extended abstract.

Obstacles

• Programming in Beluga is hard.

Obstacles

• Programming in Beluga is hard. Like, really hard.

Obstacles

- Programming in Beluga is hard. Like, really hard.
- We need to first understand the proofs for System F and $\lambda \mathbf{P}$.