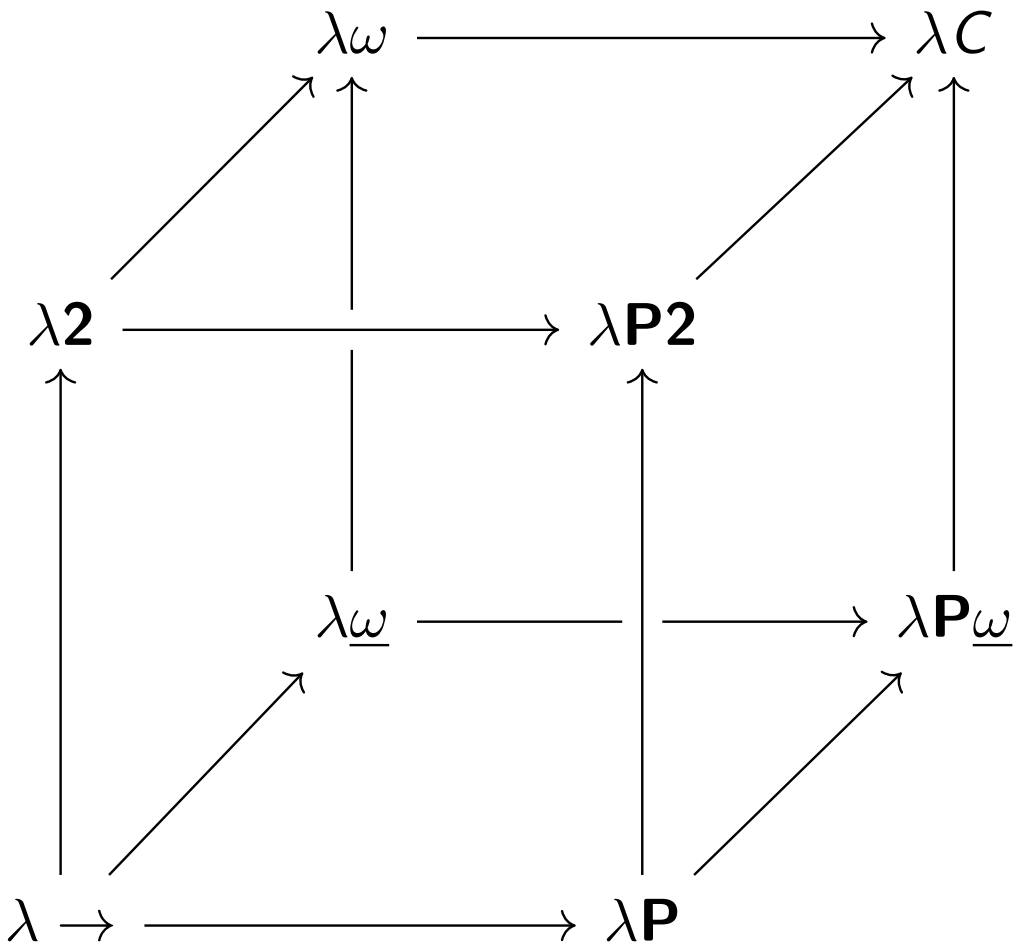
527 Presentation

_, _, _, Milton Rosenbaum

Comp 527

2025

The Cube


$\lambda\omega$
λC
$\lambda\mathbf{2}$
$\lambda\mathbf{P2}$
$\lambda\underline{\omega}$
$\lambda\mathbf{P}\underline{\omega}$
$\lambda\rightarrow$
$\lambda\mathbf{P}$

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 **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

$(\star, \star)$	Terms can depend on terms
$(\Box, \star)$	Terms can depend on types
$(\star, \Box)$	Types can depend on terms
$(\Box, \Box)$	Types can depend on types

Parameterized & Homogeneous

$\lambda\rightarrow$	$(\star,\star)$			
$\lambda\mathbf{2}$	$(\star,\star)$	$(\Box,\star)$		
$\lambda\underline{\omega}$	$(\star,\star)$		$(\Box,\Box)$	
$\lambda\omega$	$(\star,\star)$	$(\Box,\star)$	$(\Box,\Box)$	
$\lambda\mathbf{P}$	$(\star,\star)$			$(\star,\Box)$
$\lambda\mathbf{P2}$	$(\star,\star)$	$(\Box,\star)$		$(\star,\Box)$
$\lambda\mathbf{P}\underline{\omega}$	$(\star,\star)$		$(\Box,\Box)$	$(\star,\Box)$
$\lambda\mathbf{P}\omega$	$(\star,\star)$	$(\Box,\star)$	$(\Box,\Box)$	$(\star,\Box)$

Just change two two rules!

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

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

What are they

In addition to $\Box$ and $\star$, we can add arbitrary sorts and axiom relations. A GTS, $\lambda(S, A, R)$ can be described as

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

What are they

In addition to $\Box$ and $\star$, we can add arbitrary sorts and axiom relations. A GTS, $\lambda(S, A, R)$ can be described as

- S : a set of sorts, for example $\star, \Box$
- A : a set of axioms, $\star : \Box$

What are they

In addition to $\Box$ and $\star$, we can add arbitrary sorts and axiom relations. A GTS, $\lambda(S, A, R)$ can be described as

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

Even more systems

A surprising number of systems can be defined as PTSs:

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

Even more systems

A surprising number of systems can be defined as PTSs:

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

Mechanize Proof for Each PTS

since so many systems are easily describable as GTSs, it would be nice to prove some properties about the systems in general.
In our project, we plan to:

- Mechanize Arbitrary PTSs in Beluga

Mechanize Proof for Each PTS

since so many systems are easily describable as GTSs, it would be nice to prove some properties about the systems in general.
In our project, we plan to:

- Mechanize Arbitrary PTSs in Beluga
- Prove Preservation and Progress for a generalized PTS

Tasks

- Milton: Implementing Generalized PTSs in Beluga, mechanizing the proofs

Tasks

- Milton: Implementing Generalized PTSs in Beluga, mechanizing the proofs
- Everyone else: Everything else!

Obstacles

- Programming in Beluga is *hard*.

Obstacles

- Programming in Beluga is *hard*.
- Like, really hard.