

propositional logic & simple types

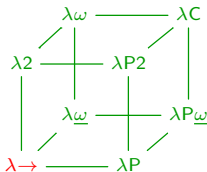
Freek Wiedijk

Type Theory & Coq

2024–2025

Radboud University Nijmegen

September 13, 2024



type systems and logics

teaser: a proof term for a proof

$\lambda x : a \rightarrow b \rightarrow c. \lambda y : a \rightarrow b. \lambda z : a. xz(yz)$

$$\frac{\frac{\frac{[a \rightarrow b \rightarrow c^x] \quad [a^z]}{b \rightarrow c} E \rightarrow \quad \frac{\frac{[a \rightarrow b^y] \quad [a^z]}{b} E \rightarrow}{c} E \rightarrow}{\frac{c}{a \rightarrow c} I[z] \rightarrow} I[y] \rightarrow \frac{(a \rightarrow b) \rightarrow a \rightarrow c}{(a \rightarrow b \rightarrow c) \rightarrow (a \rightarrow b) \rightarrow a \rightarrow c} I[x] \rightarrow$$

many systems

- ▶ one untyped lambda calculus
(last week: recap)
- ▶ many typed lambda calculi = type theories
 - ▶ **Church-style**
variables explicitly typed
the systems in this course
Coq!
 - ▶ **Curry-style**
assigning types to untyped terms
(in one of Herman's lectures)
- ▶ many logics

Curry-Howard correspondence

'propositions-as-types'

type systems \sim logics

datatypes \sim propositions

$A \times B \sim A \wedge B$

$A \rightarrow B \sim A \rightarrow B$

objects of a datatype \sim proofs of a proposition
'inhabitants' \sim 'proof objects'

non-empty type \sim true proposition

empty type \sim false proposition

derivation in a type theory \sim derivation in a logic

type systems in this course

logics \sim type systems

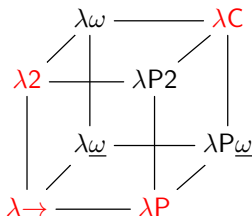
propositional logic $\sim \lambda \rightarrow$ = simply typed lambda calculus

predicate logic $\sim \lambda P$ = dependently typed lambda calculus

second order logic $\sim \lambda 2$ = polymorphic lambda calculus

higher order logic $\sim \lambda C$ = CC = Calculus of Constructions

'the logic of Coq' \sim CIC = Calculus of Inductive Constructions

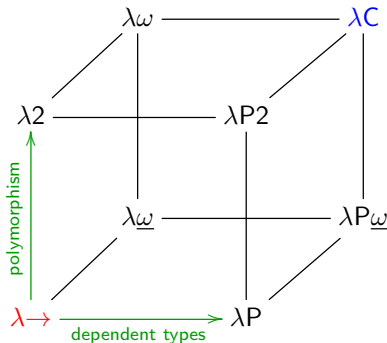


the lambda cube

the Barendregt cube

eight PTSs = pure type systems

(not explicitly in Femke's course notes)



► **natural deduction**

$$B_1, \dots, B_m \vdash A$$

introduction and elimination rules for each connective

$$\frac{\dots \vdash \dots}{\dots \vdash \dots \otimes \dots} I$$

$$\frac{\dots \vdash \dots \otimes \dots}{\dots \vdash \dots} E$$

- **Gentzen–Prawitz style:** proof is a derivation tree
- **Jaśkowski–Fitch style:** proof consists of nested boxes/flags

► **sequent calculus:** LK & LJ

$$B_1, \dots, B_m \vdash A_1, \dots, A_n$$

left and right rules for each connective

$$\frac{\dots \vdash \dots}{\dots \otimes \dots \vdash \dots} L$$

$$\frac{\dots \vdash \dots}{\dots \vdash \dots \otimes \dots} R$$

► **Hilbert-style**

$$\vdash A$$

axioms for each connective (and at most two rules)

defining a logic or type system

► syntax

- terms
- types
- formulas
- contexts
- judgments

$$M, N ::= x \mid MN \mid \lambda x. M$$

► rules

- **logics**: proof rules
- **type systems**: typing rules

no rules for untyped lambda calculus

► reduction

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

propositional logic

three propositional logics

- ▶ **minimal propositional logic**

the logic that corresponds to simply typed lambda calculus

only implication: \rightarrow

- ▶ **constructive propositional logic**

all connectives: $\rightarrow, \wedge, \vee, \neg, \perp, \top$

- ▶ **classical propositional logic**

$$\begin{aligned} & A \vee \neg A \\ & \neg\neg A \rightarrow A \end{aligned}$$

propositional logic: syntax

formulas

$$A, B ::= a \mid A \rightarrow B \mid A \wedge B \mid A \vee B \mid \neg A \mid \top \mid \perp$$

a, b, c, \dots atomic propositions
later: propositional variables

A, B, C, \dots meta-variables for arbitrary formulas

implication associates to the right:

$$a \rightarrow b \rightarrow c \text{ means } a \rightarrow (b \rightarrow c)$$

order of binding strength: $\neg > \wedge > \vee > \rightarrow$

$$a \vee \neg b \wedge c \rightarrow d \text{ means } (a \vee ((\neg b) \wedge c)) \rightarrow d$$

the two rules of minimal propositional logic

the introduction and elimination rules of implication:

$$\frac{\begin{array}{c} [A^x] \\ \vdots \\ B \end{array}}{A \rightarrow B} I[x] \rightarrow \qquad \frac{\begin{array}{c} \vdots \\ A \rightarrow B \end{array} \quad \begin{array}{c} \vdots \\ A \end{array}}{B} E \rightarrow$$

in the introduction rule the assumption $[A^x]$ may be used an arbitrary number of times: zero, one or more

in the elimination rule the proof of $A \rightarrow B$ is on the *left* of the proof of A

example proof of minimal propositional logic

$$\frac{\frac{\frac{[a \rightarrow b \rightarrow c^x] \quad [a^z]}{b \rightarrow c} E \rightarrow \quad \frac{\frac{[a \rightarrow b^y] \quad [a^z]}{b} E \rightarrow}{c} I[z] \rightarrow}{\frac{a \rightarrow c}{(a \rightarrow b) \rightarrow a \rightarrow c} I[y] \rightarrow} I[x] \rightarrow$$
$$(a \rightarrow b \rightarrow c) \rightarrow (a \rightarrow b) \rightarrow a \rightarrow c$$

constructive propositional logic: the other rules

$$\begin{array}{c}
\vdots \quad \vdots \\
\frac{A \quad B}{A \wedge B} I\wedge \qquad \frac{\vdots \quad A \wedge B}{A} El\wedge \qquad \frac{\vdots \quad A \wedge B}{B} Er\wedge \\
\\
\vdots \quad \vdots \qquad \vdots \quad \vdots \quad \vdots \\
\frac{A}{A \vee B} Il\vee \qquad \frac{B}{A \vee B} Ir\vee \qquad \frac{\vdots \quad A \vee B \quad \vdots \quad A \rightarrow C \quad \vdots \quad B \rightarrow C}{C} EV \\
\\
[A^x] \\
\vdots \quad \vdots \quad \vdots \quad \vdots \\
\frac{\perp}{\neg A} I\neg \qquad \frac{\neg A \quad A}{\perp} E\neg \qquad \frac{}{\top} I\top \qquad \frac{\vdots \quad \perp}{A} E\perp \\
\\
\neg A := A \rightarrow \perp
\end{array}$$

variants of elimination rules

- what Coq's `elim` tactic implements for conjunction:

$$\frac{\begin{array}{c} \vdots \\ A \wedge B \end{array} \quad \begin{array}{c} \vdots \\ A \rightarrow B \rightarrow C \end{array}}{C} E\wedge$$

- often the disjunction elimination rule is:

$$\frac{\begin{array}{c} \vdots \\ A \vee B \end{array} \quad \begin{array}{c} [A^x] \\ \vdots \\ C \end{array} \quad \begin{array}{c} [B^y] \\ \vdots \\ C \end{array}}{C} E[x, y]\vee$$

not what Coq's `elim` tactic implements for disjunction

example proof beyond minimal propositional logic

$$\frac{[a \vee b^x] \quad \frac{\frac{\frac{[a^y]}{b \vee a} Ir\vee}{a \rightarrow b \vee a} I[y] \rightarrow \quad \frac{\frac{\frac{[b^z]}{b \vee a} Il\vee}{b \rightarrow b \vee a} I[z] \rightarrow}{E\vee}}{b \vee a} \quad \frac{b \vee a}{a \vee b \rightarrow b \vee a} I[x] \rightarrow$$

alternative style: explicit assumption lists

syntax

$A, B ::= a \mid A \rightarrow B \mid \dots$	formulas
$\Gamma ::= \cdot \mid \Gamma, A$	assumption lists
$\mathcal{J} ::= \Gamma \vdash A$	sequents

we do not write the dot and the comma after the dot
still natural deduction, *not* sequent calculus

rules

$$\frac{}{\Gamma \vdash A} \text{ass} \quad \text{for } A \in \Gamma$$
$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \rightarrow B} I \rightarrow \quad \frac{\Gamma \vdash A \rightarrow B \quad \Gamma \vdash A}{\Gamma \vdash B} E \rightarrow \quad \dots$$

the example proofs with explicit assumption lists

$\Gamma_1 := a \rightarrow b \rightarrow c, a \rightarrow b, a$

$$\begin{array}{c}
 \frac{}{\Gamma_1 \vdash a \rightarrow b \rightarrow c} \text{ass} \quad \frac{}{\Gamma_1 \vdash a} \text{ass} \quad \frac{}{\Gamma_1 \vdash a \rightarrow b} \text{ass} \quad \frac{}{\Gamma_1 \vdash a} \text{ass} \\
 \frac{}{\Gamma_1 \vdash b \rightarrow c} E \rightarrow \quad \frac{}{\Gamma_1 \vdash b} E \rightarrow \\
 \frac{}{\Gamma_1 \vdash c} E \rightarrow \\
 \frac{}{a \rightarrow b \rightarrow c, a \rightarrow b \vdash a \rightarrow c} I \rightarrow \\
 \frac{}{a \rightarrow b \rightarrow c \vdash (a \rightarrow b) \rightarrow a \rightarrow c} I \rightarrow \\
 \frac{}{\vdash (a \rightarrow b \rightarrow c) \rightarrow (a \rightarrow b) \rightarrow a \rightarrow c} I \rightarrow
 \end{array}$$

$$\begin{array}{c}
 \frac{}{a \vee b \vdash a \vee b} \text{ass} \quad \frac{}{a \vee b, a \vdash a} \text{ass} \quad \frac{}{a \vee b, b \vdash b} \text{ass} \\
 \frac{}{a \vee b, a \vdash b \vee a} Ir \vee \quad \frac{}{a \vee b, b \vdash b \vee a} Il \vee \\
 \frac{}{a \vee b \vdash a \rightarrow b \vee a} I \rightarrow \quad \frac{}{a \vee b \vdash b \rightarrow b \vee a} I \rightarrow \\
 \frac{}{a \vee b \vdash b \vee a} E \vee \\
 \frac{}{\vdash a \vee b \rightarrow b \vee a} I \rightarrow
 \end{array}$$

simply typed lambda calculus

the S combinator

untyped

typed: Curry-style

$$\lambda xyz. xz(yz)$$
$$(\lambda x. (\lambda y. (\lambda z. ((xz)(yz)))))$$

typed: Church-style

$$\lambda x : a \rightarrow b \rightarrow c. \lambda y : a \rightarrow b. \lambda z : a. xz(yz)$$

Coq syntax

```
fun x : a -> b -> c => fun y : a -> b => fun z : a =>
  x z (y z)
```

```
fun (x : a -> b -> c) (y : a -> b) (z : a) => x z (y z)
```

BHK interpretation

Brouwer–Heyting–Kolmogorov

~ Curry-Howard correspondence

constructive ‘meaning’ of the logical connectives

connection between proofs and lambda calculus

proof of $A \rightarrow B$	=	function from proofs of A to proofs of B
proof of $A \wedge B$	=	pair of a proof of A and a proof of B
proof of $A \vee B$	=	either a proof of A , or a proof of B
proof of $\neg A$	=	proof of $A \rightarrow \perp$
proof of \top		the object I
proof of \perp		does not exist

proof of $a \wedge b \rightarrow b \wedge a$ is

a function that inputs a pair $\langle x, y \rangle$, and returns $\langle y, x \rangle$
with x a proof of a and y a proof of b

$$\lambda p : a \wedge b. \langle \pi_2 p, \pi_1 p \rangle$$

different names for the same theory

simply typed lambda calculus

\parallel

simple type theory = STT

\parallel

$\lambda \rightarrow$

3 typing rules

\nVdash

7 typing rules

same set of well-typed terms

simply typed lambda calculus: syntax and rules

syntax

$A, B ::= a \mid A \rightarrow B$	types
$M, N ::= x \mid MN \mid \lambda x : A. M$	preterms
$\Gamma ::= \cdot \mid \Gamma, x : A$	contexts
$\mathcal{J} ::= \Gamma \vdash M : A$	judgments

rules

$$\frac{}{\Gamma \vdash x : A} \quad \text{for } (x : A) \in \Gamma$$
$$\frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash (\lambda x : A. M) : A \rightarrow B} \quad \frac{\Gamma \vdash F : A \rightarrow B \quad \Gamma \vdash M : A}{\Gamma \vdash FM : B}$$

well-typedness

$\Gamma \vdash M : A$ is derivable $\implies M$ is well-typed

well-typed preterms are called terms

example type derivation

$\lambda f : a \rightarrow b. \lambda x : a. fx$

$$\frac{\frac{\frac{f : a \rightarrow b, x : a \vdash f : a \rightarrow b}{f : a \rightarrow b, x : a \vdash fx : b}}{f : a \rightarrow b \vdash (\lambda x : a. fx) : a \rightarrow b}}{\vdash (\lambda f : a \rightarrow b. \lambda x : a. fx) : (a \rightarrow b) \rightarrow a \rightarrow b}$$

$$\frac{}{\Gamma \vdash x : A} \quad \frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash (\lambda x : A. M) : A \rightarrow B} \quad \frac{\Gamma \vdash F : A \rightarrow B \quad \Gamma \vdash M : A}{\Gamma \vdash FM : B}$$

proofs versus type derivations: type derivation

$$\begin{array}{c}
 \frac{\frac{}{f : a \rightarrow b, x : a \vdash f : a \rightarrow b} \quad \frac{}{f : a \rightarrow b, x : a \vdash x : a}}{\frac{}{f : a \rightarrow b, x : a \vdash fx : b}} \\
 \frac{}{f : a \rightarrow b \vdash (\lambda x : a. fx) : a \rightarrow b} \\
 \hline
 \vdash (\lambda f : a \rightarrow b. \lambda x : a. fx) : (a \rightarrow b) \rightarrow a \rightarrow b
 \end{array}$$

$$\begin{array}{c}
 \frac{}{a \rightarrow b, a \vdash a \rightarrow b} \text{ass} \quad \frac{}{a \rightarrow b, a \vdash a} \text{ass} \\
 \hline
 \frac{}{a \rightarrow b, a \vdash b} E \rightarrow \\
 \frac{}{a \rightarrow b, a \vdash b} I \rightarrow \\
 \frac{}{a \rightarrow b \vdash a \rightarrow b} I \rightarrow \\
 \hline
 \vdash (a \rightarrow b) \rightarrow a \rightarrow b
 \end{array}$$

proofs versus type derivations: proof

$$\begin{array}{c}
 \frac{[a \rightarrow b^f] \quad [a^x]}{b} E \rightarrow \\
 \frac{a \rightarrow b}{a \rightarrow b} I[x] \rightarrow \\
 \frac{}{(a \rightarrow b) \rightarrow a \rightarrow b} I[f] \rightarrow
 \end{array}$$

$$\begin{array}{c}
 \frac{}{a \rightarrow b, a \vdash a \rightarrow b} \text{ass} \quad \frac{}{a \rightarrow b, a \vdash a} \text{ass} \\
 \frac{}{a \rightarrow b, a \vdash b} E \rightarrow \\
 \frac{a \rightarrow b, a \vdash b}{a \rightarrow b \vdash a \rightarrow b} I \rightarrow \\
 \frac{a \rightarrow b \vdash a \rightarrow b}{\vdash (a \rightarrow b) \rightarrow a \rightarrow b} I \rightarrow
 \end{array}$$

proofs versus type derivations: proof term

$$\frac{\frac{\frac{[a \rightarrow b^f] \quad [a^x]}{b} E \rightarrow}{a \rightarrow b} I[x] \rightarrow}{(a \rightarrow b) \rightarrow a \rightarrow b} I[f] \rightarrow$$

$$\lambda f : a \rightarrow b. \lambda x : a. f x$$

Curry-Howard correspondence

propositions

types

implications \longleftrightarrow function types

proof rules

proof terms

introduction rules \longleftrightarrow lambda abstraction

elimination rules \longleftrightarrow function application

assumption rule \longleftrightarrow variables

how to read proof terms

$$M := (\lambda f : a \rightarrow b. \lambda x : a. fx) : (a \rightarrow b) \rightarrow a \rightarrow b$$

this proof term M is a function
that takes as its first argument a proof of $a \rightarrow b$ called f
and as its second argument a proof of a called x
and then maps those to a proof of b

the argument f is itself a function that
maps proofs of a to proofs of b
(conform the BHK-interpretation)

x is an inhabitant of the type a
which corresponds to the proposition a

to get a proof of b , the function f is applied to x
and the result of that application then is the result of M

Coq

the example

```
Parameter a b : Prop.      one = fun (f : a -> b) (x : a) => f x
                             : (a -> b) -> a -> b
```

```
Lemma one :
  (a -> b) -> a -> b.
intros f x.
apply f.
apply x.
Qed.
```

```
Check one.
Print one.
```

$$\frac{\frac{\frac{[a \rightarrow b^f] \quad [a^x]}{b} E \rightarrow}{a \rightarrow b} I[x] \rightarrow}{(a \rightarrow b) \rightarrow a \rightarrow b} I[f] \rightarrow$$

$$(\lambda f : a \rightarrow b. \lambda x : a. fx) : (a \rightarrow b) \rightarrow a \rightarrow b$$

example with disjunction

Parameter a b : Prop.

Lemma two :

$a \vee b \rightarrow b \vee a$.

intros [y|z].

- right.

 apply y.

- left.

 apply z.

Qed.

$$\frac{\frac{[a^y]}{b \vee a} Ir\vee \quad \frac{[b^z]}{b \vee a} Il\vee}{\frac{[a \vee b^x]}{a \rightarrow b \vee a} I[y] \rightarrow \quad \frac{b \rightarrow b \vee a}{I[z] \rightarrow} E\vee} \frac{b \vee a}{a \vee b \rightarrow b \vee a} I[x] \rightarrow$$

example with conjunction

Parameter a b : Prop.

Lemma three :

$a \wedge b \rightarrow b \wedge a$.

intros [y z].

split.

- apply z.

- apply y.

Qed.

$$\frac{\frac{\frac{[b^z] \quad [a^y]}{b \wedge a} I \wedge \quad \frac{b \wedge a}{b \rightarrow b \wedge a} I[z] \rightarrow \quad \frac{b \rightarrow b \wedge a}{a \rightarrow b \rightarrow b \wedge a} I[y] \rightarrow}{[a \wedge b^x] \quad a \rightarrow b \rightarrow b \wedge a} E \wedge}{b \wedge a} \frac{b \wedge a}{a \wedge b \rightarrow b \wedge a} I[x] \rightarrow$$

tactics for proof rules

$I \rightarrow I \neg$	intro intros	
$E \rightarrow E \neg$	apply	
ass	exact apply	
$E \wedge E \vee E \perp$	elim destruct intros with pattern	
$I \wedge$	split	
$Il \vee$	left	
$Ir \vee$	right	
$I \top$	apply I	I : True

bullets

structure tactic scripts according to subgoals

related bullets need to match:

- -- --- ---- etc.

+ ++ +++ ++++ etc.

* ** *** **** etc.

intro patterns

only works with `intros`, not with `intro`

the pattern needs to match the shape of assumptions

goal	tactic
$A \rightarrow G$	<code>intros HA.</code>
$A \rightarrow B \rightarrow G$	<code>intros HA HB.</code>
$A \wedge B \rightarrow G$	<code>intros [HA HB].</code>
$A \vee B \rightarrow G$	<code>intros [HA HB].</code>
$A \vee (B \wedge C) \rightarrow D \rightarrow G$	<code>intros [HA [HB HC]] HD.</code>

apply

the number n of antecedents of H may be zero
 H not only a variable, may be an arbitrary term

$$H : A_1 \rightarrow \cdots \rightarrow A_n \rightarrow B$$

goal B $\xrightarrow{\text{apply } H}$ new goals A_1, \dots, A_n

$$\begin{array}{c} \frac{[A_1 \rightarrow \cdots \rightarrow A_n \rightarrow B^H] \quad A_1}{A_2 \rightarrow \cdots \rightarrow A_n \rightarrow B} E \rightarrow \\ \hline \frac{A_2 \rightarrow \cdots \rightarrow A_n \rightarrow B \quad A_2}{A_3 \rightarrow \cdots \rightarrow A_n \rightarrow B} E \rightarrow \\ \vdots \\ \frac{A_{n-1} \rightarrow A_n \rightarrow B \quad A_{n-1}}{A_n \rightarrow B} E \rightarrow \\ \hline B \quad A_n \quad E \rightarrow \end{array}$$

elim

$$H : A_1 \rightarrow \cdots \rightarrow A_n \rightarrow B \otimes C$$

`elim` and `destruct` eliminate the connective \otimes
which leaves the goal unchanged

but also generate extra subgoals A_1, \dots, A_n

summary

logic	type theory	Coq
introduction rules	lambda abstraction	<code>intro</code>
elimination rules	function application	<code>apply</code>
introduction rules	(in three weeks)	(various)
elimination rules	(in three weeks)	<code>elim</code>

for more about the tactics
read the Coq manual!



table of contents

contents

type systems and logics

propositional logic

simply typed lambda calculus

Coq

table of contents