Curry-Howard Correspondance

Jean-Pierre Jouannaud
Project Formes
INRIA-LIAMA and Tsinghua University

2nd Asian-Pacific School on Formal Methods, August 20, 2010

Outline

1. Natural deduction proofs for minimal logic
2. Functional interpretation of natural deduction
3. Proof terms
4. First-order logic
5. Induction

Outline

Outline

1. Natural deduction proofs for minimal logic
2. Functional interpretation of natural deduction
3. Proof terms
4. First-order logic
5. Induction

Outline

1. Natural deduction proofs for minimal logic
2. Functional interpretation of natural deduction
3. Proof terms
4. First-order logic
5. Induction

Outline

1. Natural deduction proofs for minimal logic
2. Functional interpretation of natural deduction
3. Proof terms
4. First-order logic
5. Induction

Outline

Minimal logic is the fragment of propositionnal logic with implication as single connective:

$$[\textsc{Intro}]\,\frac{\Gamma, A \vdash B}{\Gamma \vdash A \to B} \qquad \frac{\Gamma \vdash A \qquad \Gamma \vdash A \to B}{\Gamma \vdash B}\,[\textsc{Elim}]$$

$$\frac{A \in \Gamma}{\Gamma \vdash A}\,[\textsc{Axiom}]$$

where the *environment* Γ is a set of formulae taken as *assumptions*.

Example of proof: $\vdash A \to (A \to B) \to B$

$$
\dfrac{\dfrac{\dfrac{\dfrac{A \in A, A \to B}{A, A \to B \vdash A}[A] \qquad \dfrac{A \to B \in A, A \to B}{A, A \to B \vdash A \to B}[A]}{A, A \to B \vdash B}[E]}{A \vdash (A \to B) \to B}[I]}{\vdash A \to ((A \to B) \to B)}
$$

Remark 1: a poof is a tree which root is on the bottom and the leaves in the air.

Remark 2: *there are two essential readings for a proof:*

1. from top to bottom: forward proof
2. from bottom to top: backward proof

Example of proof: $\vdash A \to (A \to B) \to B$

$$
\dfrac{\dfrac{\dfrac{\dfrac{A \in A, A \to B}{A, A \to B \vdash A}[A] \qquad \dfrac{A \to B \in A, A \to B}{A, A \to B \vdash A \to B}[A]}{A, A \to B \vdash B}[E]}{A \vdash (A \to B) \to B}[I]}{\vdash A \to ((A \to B) \to B)}
$$

Remark 1: a poof is a tree which root is on the bottom and the leaves in the air.

Remark 2: *there are two essential readings for a proof:*

1. from top to bottom: forward proof
2. from bottom to top: backward proof

Outline

Meaning of arrow introduction

$$[\textsc{Intro}]\ \frac{\Gamma, A \vdash B}{\Gamma \vdash A \to B}$$

Arrow introduction yields a proof of $A \to B$
from a proof of B obtained by assumming a proof of A:
the proof of $A \to B$ differs of the proof of B by abstracting over all possible proofs of A.

A proof of $A \to B$ is therefore a function waiting for its argument, a proof of A, in order to return a *proof of B*.

Meaning of arrow introduction

$$[\textsc{Intro}]\ \frac{\Gamma, A \vdash B}{\Gamma \vdash A \to B}$$

Arrow introduction yields a proof of $A \to B$
from a proof of B obtained by assumming a proof of A:
the proof of $A \to B$ differs of the proof of B by abstracting over all possible proofs of A.

A proof of $A \to B$ is therefore a function waiting for its argument, a proof of A, in order to return a *proof of B*.

Meaning of arrow introduction

$$[\textsc{Intro}]\ \frac{\Gamma, A \vdash B}{\Gamma \vdash A \to B}$$

Arrow introduction yields a proof of $A \to B$
from a proof of B obtained by assumming a proof of A:
the proof of $A \to B$ differs of the proof of B by abstracting over all possible proofs of A.

A proof of $A \to B$ is therefore a function waiting for its argument, a proof of A, in order to return a *proof of B*.

Meaning of arrow elimination

$$[\text{ELIM}]\ \frac{\Gamma \vdash A \to B \qquad \Gamma \vdash A}{\Gamma \vdash B}$$

Arrow elimination yields a proof of B
from a proof of $A \to B$ and a proof of A:

a proof of B is obtained by *application* of a proof of $A \to B$ (a function) to an actual *proof of A* (the argument).

Meaning of axiom

$$[\textsc{Axiom}]\ \frac{A \in \Gamma}{\Gamma \vdash A}$$

Axiom returns the given proof of A:

it is the *identity*.

Outline

Proofs as objects in a programming language

To this end, we need a language in which to

- express functions and function application
- check that an argument is appropriate

Such a language exists already, it is

(SIMPLY) TYPED LAMBDA CALCULUS

Proofs as objects in a programming language

To this end, we need a language in which to

- express functions and function application
- check that an argument is appropriate

Such a language exists already, it is

(SIMPLY) TYPED LAMBDA CALCULUS

Proof term for arrow introduction

$$\frac{A \vdash \qquad B}{\vdash \qquad\qquad A \to B}\;[\textsc{Intro}]$$

Arrow introduction has two readings:

- A logical reading:
  If u is a proof term for B under the assumption that x names an arbitrary proof term for A, then $\lambda x : A.u$ is a proof term for $A \to B$.
- A computational reading:
  If u has type B in the environment in which x has type A, then $\lambda x : A.u$ has type $A \to B$.
- Note that assumptions become pairs made of a proof name and a formula.

Proof term for arrow introduction

$$\frac{x : A \vdash u : B}{\vdash (\lambda x : A.u) : A \to B}\ [\textsc{Intro}]$$

Arrow introduction has two readings:

- A logical reading:
  If u is a proof term for B under the assumption that x names an arbitrary proof term for A, then $\lambda x : A.u$ is a proof term for $A \to B$.
- A computational reading:
  If u has type B in the environment in which x has type A, then $\lambda x : A.u$ has type $A \to B$.
- Note that assumptions become pairs made of a proof name and a formula.

Proof term for arrow introduction

$$\frac{x : A \vdash u : B}{\vdash (\lambda x : A.u) : A \to B}\ [\textsc{Intro}]$$

Arrow introduction has two readings:

- A logical reading:
  If u is a proof term for B under the assumption that x names an arbitrary proof term for A, then $\lambda x : A.u$ is a proof term for $A \to B$.
- A computational reading:
  If u has type B in the environment in which x has type A, then $\lambda x : A.u$ has type $A \to B$.
- Note that assumptions become pairs made of a proof name and a formula.

Proof term for arrow introduction

$$\frac{x: \ A \vdash u: \ B}{\vdash (\lambda x : A.u): \ A \to B} \ [\textsc{Intro}]$$

Arrow introduction has two readings:

- A logical reading:
  If u is a proof term for B under the assumption that x names an arbitrary proof term for A, then $\lambda x : A.u$ is a proof term for $A \to B$.
- A computational reading:
  If u has type B in the environment in which x has type A, then $\lambda x : A.u$ has type $A \to B$.
- Note that assumptions become pairs made of a proof name and a formula.

Proof term for arrow introduction

$$\frac{x : A \vdash u : B}{\vdash (\lambda x : A.u) : A \to B}\,[\textsc{Intro}]$$

Arrow introduction has two readings:

- A logical reading:
  If u is a proof term for B under the assumption that x names an arbitrary proof term for A, then $\lambda x : A.u$ is a proof term for $A \to B$.
- A computational reading:
  If u has type B in the environment in which x has type A, then $\lambda x : A.u$ has type $A \to B$.
- Note that assumptions become pairs made of a proof name and a formula.

Proof term for arrow introduction

$$\frac{x : A \vdash u : B}{\vdash (\lambda x : A.u) : A \to B} \text{[INTRO]}$$

Arrow introduction has two readings:

- A logical reading:
  If u is a proof term for B under the assumption that x names an arbitrary proof term for A, then $\lambda x : A.u$ is a proof term for $A \to B$.
- A computational reading:
  If u has type B in the environment in which x has type A, then $\lambda x : A.u$ has type $A \to B$.
- Note that assumptions become pairs made of a proof name and a formula.

Proof term for arrow elimination

$$\frac{\vdash \quad A \to B \qquad \vdash \quad B}{\vdash \qquad B}\ [\textsc{Elim}]$$

Arrow elimination has two readings:

- A logical reading:
  If u is a proof term for $A \to B$ and v a proof term for A, then $(u\,v)$ is a proof term for B.
- A computational reading:
  If u has type $A \to B$ and v has type A, then $(u\,v)$ has type B.

Proof term for arrow elimination

$$\frac{\vdash u : A \to B \qquad \vdash v : B}{\vdash (u\,v) : B}\,[\textsc{Elim}]$$

Arrow elimination has two readings:

- A logical reading:
  If u is a proof term for $A \to B$ and v a proof term for A, then $(u\,v)$ is a proof term for B.
- A computational reading:
  If u has type $A \to B$ and v has type A, then $(u\,v)$ has type B.

Proof term for arrow elimination

$$\frac{\vdash u : \; A \to B \qquad \vdash v : \; B}{\vdash (u\,v) : \; B}\,[\textsc{Elim}]$$

Arrow elimination has two readings:

- A logical reading:
  If u is a proof term for $A \to B$ and v a proof term for A, then $(u\,v)$ is a proof term for B.
- A computational reading:
  If u has type $A \to B$ and v has type A, then $(u\,v)$ has type B.

Proof term for arrow elimination

$$\frac{\vdash u : A \to B \qquad \vdash v : B}{\vdash (u\,v) : B}\,[\textsc{Elim}]$$

Arrow elimination has two readings:

- A logical reading:
  If u is a proof term for $A \to B$ and v a proof term for A, then $(u\,v)$ is a proof term for B.
- A computational reading:
  If u has type $A \to B$ and v has type A, then $(u\,v)$ has type B.

Proof term for arrow elimination

$$\frac{\vdash u : A \to B \qquad \vdash v : B}{\vdash (u\,v) : B}\ [\textsc{Elim}]$$

Arrow elimination has two readings:

- A logical reading:
  If u is a proof term for $A \to B$ and v a proof term for A, then $(u\,v)$ is a proof term for B.
- A computational reading:
  If u has type $A \to B$ and v has type A, then $(u\,v)$ has type B.

Proof term for axiom

$$\frac{A \in \qquad A}{\vdash \qquad A}\ [\textsc{Axiom}]$$

Axiom has two different readings:

- A logical reading:
  If x names a proof term for A, then x is a proof term for A.
- A computational reading:
  If x is assumed to be of type A, then x has type A

Proof term for axiom

$$\frac{x : A \in x : A}{\vdash x : A}\;[\textsc{Axiom}]$$

Axiom has two different readings:

- A logical reading:
  If x names a proof term for A, then x is a proof term for A.
- A computational reading:
  If x is assumed to be of type A, then x has type A

Proof term for axiom

$$\frac{x : A \in x : A}{\vdash x : A}\;[\textsc{Axiom}]$$

Axiom has two different readings:

- A logical reading:
  If x names a proof term for A, then x is a proof term for A.
- A computational reading:
  If x is assumed to be of type A, then x has type A

Proof term for axiom

$$\frac{x : A \in x : A}{\vdash x : A}\,[\textsc{Axiom}]$$

Axiom has two different readings:

- A logical reading:
  If x names a proof term for A, then x is a proof term for A.
- A computational reading:
  If x is assumed to be of type A, then x has type A

Proof term for axiom

$$\frac{x : A \in x : A}{\vdash x : A}\ [\textsc{Axiom}]$$

Axiom has two different readings:

- A logical reading:
  If x names a proof term for A, then x is a proof term for A.
- A computational reading:
  If x is assumed to be of type A, then x has type A

Example of proof term for $\vdash A \to (A \to B) \to B$

$$
\cfrac{\cfrac{\cfrac{\cfrac{A \in A, \; A \to B}{A, \; A \to B \vdash A}\,[A] \qquad \cfrac{A \to B \in \Gamma}{\Gamma \vdash A \to B}\,[A]}{A, \; A \to B \vdash B}\,[E]}{A \vdash (A \to B) \to B}\,[I}{\vdash A \to ((A \to B) \to B)}
$$

Done

$$\dfrac{\dfrac{\dfrac{\dfrac{x : A \in x : A, y : A \to B}{x : A, y : A \to B \vdash x : A}[A] \qquad \dfrac{y : A \to B \in \Gamma}{\Gamma \vdash y : A \to B}[A]}{x : A, y : A \to B \vdash (y\,x) : B}[E]}{x : A \vdash \lambda y.((y\,x) : (A \to B) \to B}[I]}{\vdash \lambda x.\lambda y.(y\,x) : A \to ((A \to B) \to B)}$$

Done

Propositions are Types
Proofs are Programs

What about cut elimination ?

Propositions are Types
Proofs are Programs

What about cut elimination ?

Cut-elimination

$$\dfrac{\dfrac{\Gamma, \quad A \vdash \quad B}{\Gamma \vdash \quad A \to B}\,[\textsc{Intro}] \qquad \Gamma \vdash \quad A}{\Gamma \vdash \quad B}\,[\textsc{Elim}]$$

By induction on the proof term u, we get:

$$\Gamma \vdash \quad B$$

Hence, cut-elimination is functional evaluation:

$$(\lambda[x : A].u\ v) \longrightarrow_{\beta} u\{x \mapsto v\}$$

Cut-elimination

$$\dfrac{\dfrac{\Gamma, x : A \vdash u : B}{\Gamma \vdash \lambda[x : A].u : A \rightarrow B}\,[\textsc{Intro}] \qquad \Gamma \vdash v : A}{\Gamma \vdash (\lambda[x : A].u\ v) : B}\,[\textsc{Elim}]$$

By induction on the proof term u, we get:

$$\Gamma \vdash u\{x \mapsto v\} : B$$

Hence, cut-elimination is functional evaluation:

$$(\lambda[x : A].u\ v) \longrightarrow_{\beta} u\{x \mapsto v\}$$

Cut-elimination

$$\dfrac{\dfrac{\Gamma, x : A \vdash u : B}{\Gamma \vdash \lambda[x : A].u : A \to B}\,[\textsc{Intro}] \qquad \Gamma \vdash v : A}{\Gamma \vdash (\lambda[x : A].u\ v) : B}\,[\textsc{Elim}]$$

By induction on the proof term u, we get:

$$\Gamma \vdash u\{x \mapsto v\} : B$$

Hence, cut-elimination is functional evaluation:

$$(\lambda[x : A].u\ v) \longrightarrow_\beta u\{x \mapsto v\}$$

Cut-elimination

$$\dfrac{\dfrac{\Gamma, x : A \vdash u : B}{\Gamma \vdash \lambda[x : A].u : A \to B}\,[\textsc{Intro}] \qquad \Gamma \vdash v : A}{\Gamma \vdash (\lambda[x : A].u\ v) : B}\,[\textsc{Elim}]$$

By induction on the proof term u, we get:

$$\Gamma \vdash u\{x \mapsto v\} : B$$

Hence, cut-elimination is functional evaluation:

$$(\lambda[x : A].u\ v) \longrightarrow_\beta u\{x \mapsto v\}$$

Outline

Conjunction

$$\frac{\Gamma \vdash A \qquad \Gamma \vdash B}{\Gamma \vdash A \wedge B}$$

$$\frac{\Gamma \vdash A \wedge B}{\Gamma \vdash A}$$

$$\frac{\Gamma \vdash A \wedge B}{\Gamma \vdash B}$$

- Meaning of the introduction rule: a proof of $A \wedge B$ is a pair made of a proof of A and a proof of B
- Meaning of the elimination rules: a proof of A (resp. B) can be obtained from a proof of $A \wedge B$ by taking the first (resp. second) projection of the pair.

Conjunctive proof terms

$$\frac{\Gamma \vdash \quad A \qquad \Gamma \vdash \quad B}{\Gamma \vdash \qquad A \land B}\ [\textsc{Intro}]$$

$$\frac{\Gamma \vdash \quad A \land B}{\Gamma \vdash \qquad A}\ [\textsc{Elim1}] \qquad \frac{\Gamma \vdash \quad A \land B}{\Gamma \vdash \qquad B}\ [\textsc{Elim2}]$$

Conjunctive proof terms

$$\frac{\Gamma \vdash u : A \qquad \Gamma \vdash v : B}{\Gamma \vdash \langle u, v \rangle : A \wedge B} \; [\textsc{Intro}]$$

$$\frac{\Gamma \vdash w : A \wedge B}{\Gamma \vdash 1st(w) : A} \; [\textsc{Elim1}] \qquad \frac{\Gamma \vdash w : A \wedge B}{\Gamma \vdash 2nd(w) : B} \; [\textsc{Elim2}]$$

Conjunctive proof terms

$$\frac{\Gamma \vdash u : A \qquad \Gamma \vdash v : B}{\Gamma \vdash \langle u, v \rangle : A \wedge B}\,[\textsc{Intro}]$$

$$\frac{\Gamma \vdash w : A \wedge B}{\Gamma \vdash 1st(w) : A}\,[\textsc{Elim1}] \qquad \frac{\Gamma \vdash w : A \wedge B}{\Gamma \vdash 2nd(w) : B}\,[\textsc{Elim2}]$$

Cut eliminations for conjunctions

$$\dfrac{\dfrac{\Gamma \vdash A \qquad \Gamma \vdash B}{\Gamma \vdash A \wedge B}\,[\mathrm{I}]}{\Gamma \vdash A}\,[\mathrm{E}]$$

The following projection rules on proofs follow:

$$1st(< u, v >) = u \qquad 2nd(< u, v >) = v$$

Cut eliminations for conjunctions

$$\dfrac{\dfrac{\Gamma \vdash u : A \qquad \Gamma \vdash v : B}{\Gamma \vdash \; < u, v >: A \wedge B}\,[I]}{\Gamma \vdash 1st(< u, v >) : A}\,[E]$$

The following projection rules on proofs follow:

$$1st(< u, v >) = u \qquad 2nd(< u, v >) = v$$

Cut eliminations for conjunctions

$$\dfrac{\dfrac{\Gamma \vdash u : \; A \qquad \Gamma \vdash v : \; B}{\Gamma \vdash \; < u, v >: \; A \wedge B}\,[I]}{\Gamma \vdash 1st(< u, v >) : \; A}\,[E]$$

The following projection rules on proofs follow:

$$1st(< u, v >) = u \qquad 2nd(< u, v >) = v$$

The treatment of disjunction first,
and then of quantifiers
is left as a non-trivial but instructive exercise.

Outline

Natural numbers

We consider a first-order logic equiped with:

a new constant formula $\mathbf{N}$

a constant 0

a unary function symbol S

a (postfixed) membership predicate $\in \mathbf{N}$

and for expressing proof terms:

a unary function symbol P

a ternary function symbol rec

Rules for the constructors

$$\frac{}{\Gamma \vdash \quad 0 \in \mathbf{N}}\,[0]$$

$$\frac{\Gamma \vdash \quad x \in \mathbf{N}}{\Gamma \vdash \quad S(x) \in \mathbf{N}}\,[\mathrm{I}_S] \qquad \frac{\Gamma \vdash \quad S(x) \in \mathbf{N}}{\Gamma \vdash \quad x \in \mathbf{N}}\,[\mathrm{E}_S]$$

cut elimination rule: $P(S(x) \to x$

Remark: x is the cut-free proof of $x \in \mathbf{N}$ when there is no assumption of the form $S(y) \in \mathbf{N}$ from Γ used in the proof.

Rules for the constructors

$$\frac{}{\Gamma \vdash 0 : \ 0 \in \mathbf{N}} \ [0]$$

$$\frac{\Gamma \vdash x : \ x \in \mathbf{N}}{\Gamma \vdash S(x) : \ S(x) \in \mathbf{N}} \ [\mathrm{I}_S] \qquad \frac{\Gamma \vdash y : \ S(x) \in \mathbf{N}}{\Gamma \vdash P(y) : \ x \in \mathbf{N}} \ [\mathrm{E}_S]$$

cut elimination rule: $P(S(x)) \to x$

Remark: x is the cut-free proof of $x \in \mathbf{N}$ when there is no assumption of the form $S(y) \in \mathbf{N}$ from Γ used in the proof.

Rules for the constructors

$$\frac{}{\Gamma \vdash 0 : \ 0 \in \mathbf{N}} \ [0]$$

$$\frac{\Gamma \vdash x : \ x \in \mathbf{N}}{\Gamma \vdash S(x) : \ S(x) \in \mathbf{N}} \ [\mathrm{I}_S] \qquad \frac{\Gamma \vdash y : \ S(x) \in \mathbf{N}}{\Gamma \vdash P(y) : \ x \in \mathbf{N}} \ [\mathrm{E}_S]$$

cut elimination rule: $P(S(x) \to x$

Remark: x is the cut-free proof of $x \in \mathbf{N}$ when there is no assumption of the form $S(y) \in \mathbf{N}$ from Γ used in the proof.

Rules for the constructors

$$\frac{}{\Gamma \vdash 0 : \; 0 \in \mathbf{N}} \; [0]$$

$$\frac{\Gamma \vdash x : \; x \in \mathbf{N}}{\Gamma \vdash S(x) : \; S(x) \in \mathbf{N}} \; [\mathrm{I}_S] \qquad \frac{\Gamma \vdash y : \; S(x) \in \mathbf{N}}{\Gamma \vdash P(y) : \; x \in \mathbf{N}} \; [\mathrm{E}_S]$$

cut elimination rule: $\quad P(S(x) \to x$

Remark: x is the cut-free proof of $x \in \mathbf{N}$ when there is no assumption of the form $S(y) \in \mathbf{N}$ from Γ used in the proof.

Induction as an introduction rule for $\forall$

First formulation:

$$\frac{\Gamma \vdash A[0] \qquad \Gamma \vdash \forall x : \mathbb{N}.A[x] \to A[S(x)]}{\Gamma \vdash \forall x : \mathbb{N}.A} \text{[INTRO]}$$

Alternative formulation:

$$\frac{\Gamma \vdash A[0] \qquad \Gamma \vdash \forall x : \mathbb{N}.A[x] \to A[S(x)] \qquad \Gamma \vdash n \in \mathbb{N}}{\Gamma \vdash A[n]} \text{[INTRO]}$$

Meaning of the introduction rule:
A proof of $\forall x.x \in \mathbb{N} \to A$ (in short $\forall x : \mathbb{N}.A$) is a function which returns a proof of $A[n]$, when given a proof of $A[0]$, a proof of $\forall x : \mathbb{N}.A[x] \to A[S(x)]$, and a natural number n,

We therefore need a function symbol with three arguments: a natural number and two proofs, named *rec*

Induction as an introduction rule for $\forall$

First formulation:

$$\frac{\Gamma \vdash A[0] \qquad \Gamma \vdash \forall x : \mathbb{N}.A[x] \rightarrow A[S(x)]}{\Gamma \vdash \forall x : \mathbb{N}.A} \; [\textsc{Intro}]$$

Alternative formulation:

$$\frac{\Gamma \vdash A[0] \qquad \Gamma \vdash \forall x : \mathbb{N}.A[x] \rightarrow A[S(x)] \qquad \Gamma \vdash n \in \mathbb{N}}{\Gamma \vdash A[n]} \; [\textsc{Intro}]$$

Meaning of the introduction rule:

A proof of $\forall x.x \in \mathbb{N} \rightarrow A$ (in short $\forall x : \mathbb{N}.A$) is a function which returns a proof of $A[n]$, when given a proof of $A[0]$, a proof of $\forall x : \mathbb{N}.A[x] \rightarrow A[S(x)]$, and a natural number n,

We therefore need a function symbol with three arguments: a natural number and two proofs, named *rec*.

Proof terms for the induction rule

The (second form of) the introduction rule becomes:

$$\frac{\Gamma \vdash \quad A[0] \quad \Gamma \vdash \quad \forall x : \mathbb{N}.A[x] \to A[S(x)] \quad \Gamma \vdash \quad \mathbb{N}}{\Gamma \vdash \quad A[n]} \, []$$

Note: the proof rules for $n \in \mathbb{N}$ check that n is built from 0, $S()$ and integers m declared in the environment via $m \in \mathbb{N}$, hence, the proof of a valid integer is just itself.

The first form of introduction rule is therefore:

$$\frac{\Gamma \vdash \quad A[0] \quad \Gamma \vdash \quad \forall x \in \mathbb{N}.A[x] \to A[S(x)]}{\Gamma \vdash \quad \forall x : \mathbb{N}.A} \, [\text{INTRO}]$$

Proof terms for the induction rule

The (second form of) the introduction rule becomes:

$$\frac{\Gamma \vdash v : A[0] \quad \Gamma \vdash w : \forall x : \mathbb{N}.A[x] \rightarrow A[S(x)] \quad \Gamma \vdash n : \mathbb{N}}{\Gamma \vdash rec(n, v, w) : A[n]} []$$

Note: the proof rules for $n \in \mathbb{N}$ check that n is built from 0, $S()$ and integers m declared in the environment via $m \in \mathbb{N}$, hence, the proof of a valid integer is just itself.

The first form of introduction rule is therefore:

$$\frac{\Gamma \vdash v : A[0] \qquad \Gamma \vdash w : \forall x \in \mathbb{N}.A[x] \rightarrow A[S(x)]}{\Gamma \vdash \lambda x : \mathbb{N}.rec(x, v, w) : \forall x : \mathbb{N}.A} [\text{INTRO}]$$

Proof terms for the induction rule

The (second form of) the introduction rule becomes:

$$\frac{\Gamma \vdash v : \ A[0] \quad \Gamma \vdash w : \ \forall x : \mathbb{N}.A[x] \to A[S(x)] \quad \Gamma \vdash n : \ \mathbb{N}}{\Gamma \vdash rec(n, v, w) : \ A[n]} \, []$$

Note: the proof rules for $n \in \mathbb{N}$ check that n is built from 0, $S()$ and integers m declared in the environment via $m \in \mathbb{N}$, hence, the proof of a valid integer is just itself.

The first form of introduction rule is therefore:

$$\frac{\Gamma \vdash v : \ A[0] \qquad \Gamma \vdash w : \ \forall x \in \mathbb{N}.A[x] \to A[S(x)]}{\Gamma \vdash \lambda x : \mathbb{N}.rec(x, v, w) : \ \forall x : \mathbb{N}.A} \, [\textsc{Intro}]$$

Meaning and Proof term of the elimination rules

There are two elimination rules, one for 0 and one for $S()$:

$$\frac{\Gamma \vdash \quad \forall x : Nat.A}{\Gamma \vdash \quad A[0]}\ [\textsc{Elim0}]$$

$$\frac{\Gamma \vdash \quad \forall x : Nat.A \qquad \Gamma \vdash \quad n \in \mathbb{N}}{\Gamma \vdash \quad A[S(n)]}\ [\textsc{ElimS}]$$

Meaning and Proof term of the elimination rules

There are two elimination rules, one for 0 and one for $S()$:

$$\frac{\Gamma \vdash u : \forall x : Nat.A}{\Gamma \vdash (u\,0) : A[0]}\ [\textsc{Elim0}]$$

$$\frac{\Gamma \vdash u : \forall x : Nat.A \qquad \Gamma \vdash n : n \in \mathbb{N}}{\Gamma \vdash (u\,S(n)) : A[S(n)]}\ [\textsc{ElimS}]$$

There are two elimination rules, one for 0 and one for $S()$:

$$\frac{\Gamma \vdash u : \forall x : Nat.A}{\Gamma \vdash (u\,0) : A[0]}\ [\textsc{Elim}0]$$

$$\frac{\Gamma \vdash u : \forall x : Nat.A \qquad \Gamma \vdash n : n \in \mathbb{N}}{\Gamma \vdash (u\,S(n)) : A[S(n)]}\ [\textsc{Elim}S]$$

Meaning and Proof term of the elimination rules

There are two elimination rules, one for 0 and one for $S()$:

$$\frac{\Gamma \vdash u : \ \forall x : Nat.A}{\Gamma \vdash (u\,0) : \ A[0]} \ [\textsc{Elim0}]$$

$$\frac{\Gamma \vdash u : \ \forall x : Nat.A \qquad \Gamma \vdash n : \ n \in \mathbb{N}}{\Gamma \vdash (u\ S(n)) : \ A[S(n)]} \ [\textsc{ElimS}]$$

Cut elimination for [ELIM0]

$$\dfrac{\dfrac{\Gamma \vdash A[0] \qquad \Gamma \vdash \forall x : \mathbb{N}.A[x] \to A[S(x)]}{\Gamma \vdash \forall x : \mathbb{N}.A}\,[\mathrm{I}]}{\Gamma \vdash A[0]}\,[\mathrm{E0}]$$

The following rule on proofs is therefore admissible:

$$(\lambda x : \mathbb{N}.rec(x, v, w)\ 0) \to v$$

that is

$$rec(0, v, w) \to v$$

Cut elimination for [ELIM0]

$$\dfrac{\dfrac{\Gamma \vdash v : A[0] \qquad \Gamma \vdash w : \forall x : \mathbb{N}.A[x] \to A[S(x)]}{\Gamma \vdash \lambda x : \mathbb{N}.rec(x, v, w) : \forall x : \mathbb{N}.A}\,[\mathrm{I}]}{\Gamma \vdash (\lambda x : \mathbb{N}.rec(x, v, w)\ 0) : A[0]}\,[\mathrm{E0}]$$

The following rule on proofs is therefore admissible:

$$(\lambda x : \mathbb{N}.rec(x, v, w)\ 0) \to v$$

that is

$$rec(0, v, w) \to v$$

Cut elimination for [ELIM0]

$$\dfrac{\dfrac{\Gamma \vdash v : A[0] \qquad \Gamma \vdash w : \forall x : \mathbb{N}.A[x] \to A[S(x)]}{\Gamma \vdash \lambda x : \mathbb{N}.rec(x, v, w) : \forall x : \mathbb{N}.A}\,[\mathrm{I}]}{\Gamma \vdash (\lambda x : \mathbb{N}.rec(x, v, w)\ 0) : A[0]}\,[\mathrm{E0}]$$

The following rule on proofs is therefore admissible:

$$(\lambda x : \mathbb{N}.rec(x, v, w)\ 0) \to v$$

that is

$$rec(0, v, w) \to v$$

Cut elimination for [ELIMS]

$$\dfrac{\dfrac{\Gamma \vdash \quad A[0] \qquad \Gamma \vdash \quad \forall x : \mathbb{N}.A[x] \to A[S(x)]}{\Gamma \vdash \quad \forall x : \mathbb{N}.A}\,[\mathrm{I}]}{\Gamma \vdash \quad A[S(\underline{n})]}\,[\mathrm{ES}]$$

The following rule on proofs is therefore admissible:

$$(\lambda x : \mathbb{N}.rec(x, v, w)\ S(\underline{n})) \to (w\ \underline{n}\ rec(\underline{n}, v, w))$$

that is:

$$rec(S(n), v, w) \to (w\ \underline{n}\ rec(\underline{n}, v, w))$$

Cut elimination for [ELIMS]

$$\cfrac{\cfrac{\Gamma \vdash v : A[0] \qquad \Gamma \vdash w : \forall x : \mathbb{N}.A[x] \to A[S(x)]}{\Gamma \vdash \lambda x : \mathbb{N}.rec(x, v, w) : \forall x : \mathbb{N}.A}\,[\mathrm{I}]}{\Gamma \vdash (\lambda x : \mathbb{N}.rec(x, v, w)\ S(\underline{n})) : A[S(\underline{n})]}\,[\mathrm{ES}]$$

The following rule on proofs is therefore admissible:

$$(\lambda x : \mathbb{N}.rec(x, v, w)\ S(\underline{n})) \to (w\ \underline{n}\ rec(\underline{n}, v, w))$$

that is:

$$rec(S(n), v, w) \to (w\ \underline{n}\ rec(\underline{n}, v, w))$$

Cut elimination for [ELIMS]

$$\dfrac{\dfrac{\Gamma \vdash v : A[0] \qquad \Gamma \vdash w : \forall x : \mathbf{N}.A[x] \to A[S(x)]}{\Gamma \vdash \lambda x : \mathbf{N}.rec(x, v, w) : \forall x : \mathbf{N}.A}\,[\mathrm{I}]}{\Gamma \vdash (\lambda x : \mathbf{N}.rec(x, v, w)\ S(\underline{n})) : A[S(\underline{n})]}\,[\mathrm{ES}]$$

The following rule on proofs is therefore admissible:

$$(\lambda x : \mathbf{N}.rec(x, v, w)\ S(\underline{n})) \to (w\ \underline{n}\ rec(\underline{n}, v, w))$$

that is:

$$rec(S(n), v, w) \to (w\ \underline{n}\ rec(\underline{n}, v, w))$$

Gödel system T

- Logical reading: minimal logic + $\wedge$ + natural numbers generated by 0 and S + Induction.
- Computational reading: primitive recursion over natural numbers.

System T existed as a typed lambda-calculus way before the Curry-Howard isomorphism was noticed by Curry (in a format slightly more general than here).

Importance of cut elimination in logic

Because computations in (most) typed lambda calculi are terminating, any valid proposition has a cut-free proof. There are many consequences:

- Proofs are finite objects ;
- Proving consistency of a set of deductions rules: it suffices to prove that $\perp$ has no cut-free proof, which is usually easy ;
- Program extraction from proofs: cut-free proofs provide witnesses for existential quantifiers ;
- etc.

Coq: the end of the road

Coq is based on the Curry-Howard isomorphism
extended to
polymorphic types [Girard],
dependent types [De Bruijn],
their combination [Coquand],
inductive types [Coquand, Paulin-Mohring],
and a bit more ...

see:
Girard, Lafont, Taylor:
Proof and Types, Cambridge University Press, 1990.

End marker

CH-C