An idea coming from ...

- higher-order substitution (Russell, Withehead, Church, Curry, Henkin, ...)
- λ-calculus (Church, Curry, ...)
- type theory (de Bruijn, Martin-Löf, Coquand, Huet, ...)
- automated deduction (Plotkin, Peterson, Stickel, ...)
- proof-checking (Boyer, Moore, ...)
- the practice of mathematic (Appel, Haken, Hales, ...)

Proofs are built with

Deduction rules, axioms

Proofs are built with

Deduction rules, axioms and computation rules

A simple expression of this idea

In predicate logic: deduction modulo

I. Deduction modulo

II. A uniform proof language

Assumed

1. Syntax of terms and formulae in predicate logic
2. Natural deduction rules (for constructive logic)
3. Many sorted predicate logic

Deduction modulo

Proof: sequence of deduction steps

Theory: set of axioms

Deduction modulo

Proof: sequence of deduction steps and computation steps

Theory: set of axioms and computation rules

A terminating and confluent system of computation rules

Computation rules apply to terms, *e.g.*

$$0 + y \longrightarrow y$$

and to atomic formulae, *e.g.*

$$x \times y = 0 \longrightarrow x = 0 \vee y = 0$$

An example: Arithmetic

$$\forall x \ (x = x)$$

$$\forall x \ \forall y \ (x = y \Rightarrow (x/z)A \Rightarrow (y/z)A)$$

$$\forall x \forall y \ (S(x) = S(y) \Rightarrow x = y)$$

$$\forall x \ \neg 0 = S(x)$$

$$(0/z)A \Rightarrow \forall x \ ((x/z)A \Rightarrow (S(x)/z)A) \Rightarrow \forall n \ (n/z)A$$

$$\forall y \ 0 + y = y$$

$$\forall x \forall y \ S(x) + y = S(x + y)$$

$$\forall y \ 0 \times y = 0$$

$$\forall x \forall y \ S(x) \times y = x \times y + y$$

An example: Arithmetic

$$\forall x \ (x = x)$$

$$\forall x \ \forall y \ (x = y \Rightarrow (x/z)A \Rightarrow (y/z)A)$$

$$\forall x \forall y \ (S(x) = S(y) \Rightarrow x = y)$$

$$\forall x \ \neg 0 = S(x)$$

$$(0/z)A \Rightarrow \forall x \ ((x/z)A \Rightarrow (S(x)/z)A) \Rightarrow \forall n \ (n/z)A$$

$$0 + y \longrightarrow y$$

$$S(x) + y \longrightarrow S(x + y)$$

$$0 \times y \longrightarrow 0$$

$$S(x) \times y \longrightarrow x \times y + y$$

Congruence

The computation rules define a congruence on formulae *e.g.*

$$(2 \times 2 = 4) \equiv (4 = 4)$$

Smallest relation that

- is an equivalence relation
- is a congruence (compatible with all the symbols)
- contains $l \equiv r$ for each computation rule $l \longrightarrow r$

The congruence $\equiv$ is decidable (thanks to termination and confluence)

Deduction rules

Deduction rules parametrized by the congruence $\equiv$, *e.g.*

$$\frac{\Gamma \vdash A \Rightarrow B \quad \Gamma \vdash A}{\Gamma \vdash B} \Rightarrow\text{-elim}$$

$$\frac{\Gamma \vdash C \quad \Gamma \vdash A}{\Gamma \vdash B} \Rightarrow\text{-elim if } C \equiv (A \Rightarrow B)$$

How to squeeze a proof on a single slide ?

$$\dfrac{\dfrac{\dfrac{}{\forall x\ x = x \vdash \forall x\ x = x}\ \text{Axiom}}{\forall x\ x = x \vdash 2 \times 2 = 4}\ \forall\text{-elim}}{\forall x\ x = x \vdash \exists y\ 2 \times y = 4}\ \exists\text{-intro}$$

Business as usual (The equivalence lemma)

For each congruence $\equiv$, there is a theory $\mathcal{T}$ such that

$$\Gamma \vdash_{\equiv} A$$

iff

$$\mathcal{T}, \Gamma \vdash A$$

e.g. $\mathcal{T} = \{\overline{\forall}\,(P \Leftrightarrow Q) \mid P \equiv Q\}$

Nothing new from the provability point of view

Something new from the proof structure point of view

Proofs and certificates

A test: is 221 prime or composite ?

Proofs and certificates

A test: is 221 prime or composite ?

Four answers: 221 composite

- as you can check yourself
- because 13 is a divisor
- because $221 = 13 \times 17$
- because

$$\begin{array}{rrr} & \cancel{2} & \\ & 1 & 7 \\ & 1 & 3 \\ \hline & 5 & 1 \\ {}^{1}1 & 7 & \\ \hline 2 & 2 & 1 \end{array}$$

- C defined by computation rules:

$$\dfrac{}{C(221)}\ \top\text{-intro}$$

(as you can check yourself)

- $\mid$ defined by computation rules $C(x) = \exists y\ y \mid x$

$$\dfrac{\dfrac{}{13 \mid 221}\ \top\text{-intro}}{C(221)}\ \exists\text{-intro}$$

(because 13 is a divisor)

- $\times$ defined by computation rules $C(x) = \exists y \exists z\ x = y \times z$

$$\cfrac{\cfrac{\cfrac{\cfrac{}{\forall x\ (x = x)}\ \text{axiom}}{221 = 13 \times 17}\ \forall\text{-elim}}{\exists z\ (221 = 13 \times z)}\ \exists\text{-intro}}{C(221)}\ \exists\text{-intro}$$

(because $221 = 13 \times 17$)

- only axioms (each step of the computation)

$$\cfrac{\cfrac{\cfrac{\ldots}{221 = 13 \times 17}}{\exists z\ (221 = 13 \times z)}\ \exists\text{-intro}}{C(221)}\ \exists\text{-intro}$$

Purely computational theories

No axioms

Only computation rules and deduction rules

Examples: arithmetic, simple type theory, set theory

Peano fourth and fifth axioms

$$\forall x \forall y\ (S(x) = S(y) \Rightarrow x = y)$$

$$\forall x\ \neg 0 = S(x)$$

Peano fourth and fifth axioms

$$\forall x \forall y\ (S(x) = S(y) \Rightarrow x = y)$$

$$\forall x\ \neg 0 = S(x)$$

$Pred(S(x)) \longrightarrow x$

No term rule for the fourth axiom: no one point model

$Null(0) \longrightarrow \top$

$Null(S(x)) \longrightarrow \bot$

Exercise: prove the two axioms

Where are we ?

$$\forall x \ (x = x)$$

$$\forall x \ \forall y \ (x = y \Rightarrow (x/z)A \Rightarrow (y/z)A)$$

$$\forall x \forall y \ (S(x) = S(y) \Rightarrow x = y)$$

$$\forall x \ \neg 0 = S(x)$$

$$(0/z)A \Rightarrow \forall x \ ((x/z)A \Rightarrow (S(x)/z)A) \Rightarrow \forall n \ (n/z)A$$

$$\forall y \ 0 + y = y$$

$$\forall x \forall y \ S(x) + y = S(x + y)$$

$$\forall y \ 0 \times y = 0$$

$$\forall x \forall y \ S(x) \times y = x \times y + y$$

Equality

$$\forall x\ \forall y\ (x = y \Rightarrow (x/z)A \Rightarrow (y/z)A)$$

Example:

$$\forall x\ \forall y\ (x = y \Rightarrow x \leq 4 \Rightarrow y \leq 4)$$

A second sort for sets and the set $\{z \mid z \leq 4\}$: $f_{z, z \leq 4}$

$$z \in \{z \mid z \leq 4\} \Leftrightarrow z \leq 4$$

$$\forall x\ \forall y\ (x = y \Rightarrow \forall E\ (x \in E \Rightarrow y \in E))$$

Equality

$$\forall x \ \forall y \ (x = y \Rightarrow (x/z)A \Rightarrow (y/z)A)$$

Example:

$$\forall x \ \forall y \ (x = y \Rightarrow x \leq 4 \Rightarrow y \leq 4)$$

A second sort for sets and the set $\{z \mid z \leq 4\}$: $f_{z, z \leq 4}$

$$z \in \{z \mid z \leq 4\} \Leftrightarrow z \leq 4$$

$$\forall x \ \forall y \ (x = y \Leftrightarrow \forall E \ (x \in E \Rightarrow y \in E))$$

Equality

$$\forall x\ \forall y\ (x = y \Rightarrow (x/z)A \Rightarrow (y/z)A)$$

Example:

$$\forall x\ \forall y\ (x = y \Rightarrow x \leq 4 \Rightarrow y \leq 4)$$

A second sort for sets and the set $\{z \mid z \leq 4\}$: $f_{z, z \leq 4}$

$$z \in \{z \mid z \leq 4\} \longrightarrow z \leq 4$$

$$x = y \longrightarrow \forall E\ (x \in E \Rightarrow y \in E)$$

Induction

$$(0/z)A \Rightarrow \forall x \; ((x/z)A \Rightarrow (S(x)/z)A) \Rightarrow \forall n \; (n/z)A$$

Induction

$$\forall E\ (0 \in E \Rightarrow \forall x\ (x \in E \Rightarrow S(x) \in E) \Rightarrow \forall n\ n \in E)$$

Induction

$$\forall E\ (0 \in E \Rightarrow \forall x\ (x \in E \Rightarrow S(x) \in E) \Rightarrow \forall n\ (N(n) \Rightarrow n \in E))$$

Induction

$$\forall n \; (N(n) \Rightarrow \forall E \; (0 \in E \Rightarrow \forall x \; (x \in E \Rightarrow S(x) \in E) \Rightarrow n \in E))$$

Induction

$$\forall n \ (N(n) \Leftrightarrow \forall E \ (0 \in E \Rightarrow \forall x \ (x \in E \Rightarrow S(x) \in E) \Rightarrow n \in E))$$

Induction

$$N(n) \longrightarrow \forall E\ (0 \in E \Rightarrow \forall x\ (x \in E \Rightarrow S(x) \in E) \Rightarrow n \in E)$$

$$x \in f_{x,y_1,\ldots,y_n,P}(y_1,\ldots,y_n) \longrightarrow P$$

$$y = z \longrightarrow \forall E\ (y \in E \Rightarrow z \in E)$$

$$Pred(0) \longrightarrow 0 \qquad\qquad Pred(S(x)) \longrightarrow x$$

$$Null(0) \longrightarrow \top \qquad\qquad Null(S(x)) \longrightarrow \bot$$

$$N(n) \longrightarrow \forall E\ (0 \in E \Rightarrow \forall y\ (y \in E \Rightarrow S(y) \in E) \Rightarrow n \in E)$$

$$0 + y \longrightarrow y$$

$$S(x) + y \longrightarrow S(x + y)$$

$$0 \times y \longrightarrow 0$$

$$S(x) \times y \longrightarrow x \times y + y$$

I. Deduction modulo

II. A uniform proof language

a. $\lambda\Pi$

b. Axioms, non logical deduction rules, computation rules

c. An example: polymorphism

Type theories

A language to express (among other things) proofs

Proofs in which theory ?

It depends: propositional logic (simply typed λ-calculus), predicate logic ($\lambda\Pi$), arithmetic (T), second-order propositional logic (F), predicative higher-order arithmetic (ITT), second-order arithmetic (AF2), higher-order logic (CoC, Fω), full higher-order arithmetic (CIC), ...

A uniform approach ?

Representing proofs

Proof trees (2-dimensional) are tedious to draw

Data bases, communication

A useful operation on proofs: from a proof of $\Gamma, A \vdash B$ and a proof of $\Gamma \vdash A$ build a proof of $\Gamma \vdash B$

- suppress hypothesis A in all sequents
- replace axiom rules using A by the proof of $\Gamma \vdash A$

A better notation for proofs

In $A_1, ..., A_n \vdash B$, associate a variable ξ_i to each hypothesis A_i

A proof of $A_1, ..., A_n \vdash B$ = a term containing the variables $\xi_1, ..., \xi_n$

$$\frac{}{A_1, ..., A_n \vdash A_i}\ \text{Axiom}$$

ξ_i

To each rule: a function symbol (some are binders)

$$\frac{\dfrac{\pi_1}{\Gamma \vdash A} \qquad \dfrac{\pi_2}{\Gamma \vdash B}}{\Gamma \vdash A \wedge B}\ \wedge\text{-intro}$$

$f(\pi_1, \pi_2)$

$$\frac{\dfrac{\pi_1}{\Gamma, A \vdash B}}{\Gamma \vdash A \Rightarrow B}\ \Rightarrow\text{-intro}$$

$g(\xi\pi_1)$

The operation

$$\begin{array}{ccccc} \pi_2 & & \pi_2 & \pi_2 \\ & \pi_1 & & \end{array}$$

is substitution

$$(\pi_2/\xi)\pi_1$$

A notation for proofs

$$
\begin{array}{lll}
\pi ::= & & \xi \\
& | & \xi \mapsto \pi \mid (\pi_1 \ \pi_2) \qquad\qquad app(\pi_1, \pi_2) \\
& | & \langle \pi_1, \pi_2 \rangle \mid fst(\pi) \mid snd(\pi) \\
& | & i(\pi) \mid j(\pi) \mid (\delta \ \pi_1 \ \xi_1 \pi_2 \ \xi_2 \pi_3) \\
& | & I \\
& | & (\delta_\perp \ \pi) \\
& | & x \mapsto \pi \mid (\pi \ t) \\
& | & \langle t, \pi \rangle \mid (\delta_\exists \ \pi_1 \ x \xi \pi_2)
\end{array}
$$

Brouwer-Heyting-Kolmogorov interpretation of proofs

A proof of $A \wedge B$ is an ordered pair formed with a proof of A and a proof of B

A proof of $A \Rightarrow B$ is an algorithmic function mapping a proof of A to a proof of B

A proof of $\forall x\ A(x)$ is an algorithmic function mapping n to a proof of $A(n)$

Explains the notation $\xi \mapsto \pi$ and $(\pi_1\ \pi_2)$

Curry-de Bruijn-Howard isomorphism

A proof of $A \Rightarrow B$ is an algorithmic function mapping a proof of A to a proof of B

If ΦA is the type of the proofs of A then

$$\Phi(A \Rightarrow B) = \Phi A \to \Phi B$$

Φ isomorphism between formulae and types

Propositions play the role of types of their proofs

Dependent types

A proof of $\forall x\ (even(x) \vee odd(x))$ is an algorithmic function mapping n to a proof of $even(n) \vee odd(n)$

$$f(n) : even(n) \vee odd(n)$$

$$f : (x : nat) \longrightarrow (even(x) \vee odd(x))$$

$$f : \Pi x : nat\ (even(x) \vee odd(x))$$

A choice

Three languages:

$S(S(0))$ terms

$S(S(0)) = 0$ formulae

$x \mapsto x$ proofs

One language:

$S(S(0))$, $S(S(0)) = 0$ and $x \mapsto x$ are all terms of the language

One language: $\lambda\Pi$-calculus

A type T ($e.g.$ nat) for the objects of the theory

$T : Type$

One language: $\lambda\Pi$-calculus

A type T (*e.g.* nat) for the objects of the theory

$T : Type$

Translate each term as a term of type T

To each function symbol f $f : T \longrightarrow ... \longrightarrow T \longrightarrow T$

Translate each atomic formula $P(t, u)$ to a term of type $Type$

To each predicate symbol P $P : T \longrightarrow ... \longrightarrow T \longrightarrow Type$

One language: $\lambda\Pi$-calculus

A type T (*e.g.* nat) for the objects of the theory

$T : Type$

Translate each term as a term of type T

To each function symbol f $f : T \rightarrow ... \rightarrow T \rightarrow T$

Translate each atomic formula $P(t, u)$ to a term of type $Type$

To each predicate symbol P $P : T \rightarrow ... \rightarrow T \rightarrow Type$

Translate $A \Rightarrow B$ to $A \rightarrow B$

Translate $\forall x\ A(x)$ to $\Pi x : T\ A(x)$

...

An example

Assume π is a proof of $\forall x\ \forall y\ (S(x) = S(y) \Rightarrow x = y)$

And π' of $1 = 2$

Find a proof of $0 = 1$

Theories

So far: predicate logic

Need to be extended to theories (*e.g.* arithmetic, simple type theory, set theory, ...) ?

A first way to arithmetic

For each axiom: a constant

$p_3 : \forall x\ \forall y\ (S(x) = S(y) \Rightarrow x = y)$

$p_4 : \forall x\ (0 = S(x) \Rightarrow \bot)$

$Rec_A : (0/z)A \Rightarrow \forall x\ ((x/z)A \Rightarrow (S(x)/z)A) \Rightarrow \forall n\ (n/z)A$

...

$\xi : 1 = 2 \vdash (p_3\ 0\ 1\ \xi) : 0 = 1$

$\xi : 1 = 2 \vdash (p_4\ 0\ (p_3\ 0\ 1\ \xi)) : \bot$

$\vdash \xi : 1 = 2 \mapsto (p_4\ 0\ (p_3\ 0\ 1\ \xi)) : (1 = 2) \Rightarrow \bot$

A second way

Replace axioms by non logical deduction rules

$$\frac{\Gamma \vdash (0/z)A \quad \Gamma \vdash \forall x\ ((x/z)A \Rightarrow (S(x)/z)A)}{\Gamma \vdash \forall n\ (n/z)A}$$

Introduce a new construction in the proof language: $Rec(\pi_1, \pi_2)$

Almost the same but *Rec* a construction (like *app*), not a constant

A particular case, the folding and unfolding rules

$$\frac{x \in (A \cap B)}{x \in A \vee x \in B}$$

Axioms poison proof reduction

In predicate logic: a normal closed terms is an introduction

Consistency, disjunction, witness, finite failure of search of $\perp$, ...

With constants normal closed terms need not be introductions

e.g. with an axiom $\exists x\ P(x)$

No witness property, no finite failure of search of $\perp$, ...

Extra reduction rules

e.g.

$$Rec\ a\ b\ 0 \longrightarrow 0$$

$$Rec\ a\ b\ S(x) \longrightarrow (b\ x\ (Rec\ a\ b\ x))$$

ι-reduction (inductive types)

Each new axiom: a new constant and a new proof-reduction rule

Replacing axioms by computation rules: $\lambda\Pi$ modulo

$Null(0) \longrightarrow \top$

$Null(S(x)) \longrightarrow \bot$

$x \in f_{Null} \longrightarrow Null(x)$

$\forall x\ \neg(0 = S(x))$ now has the proof

$$x : nat \mapsto \alpha : (0 = S(x)) \mapsto (\alpha\ f_{Null}\ I)$$

Not a constant

No need for extra reduction rules, the terms reduces for itself

Polymorphism

Simple type theory (higher-order logic)

$$\forall P\ (P \Rightarrow P)$$

$$(Q \Rightarrow Q) \Rightarrow (Q \Rightarrow Q)$$

Polymorphism

$$\Pi P : Type\ (P \Rightarrow P)$$

built on the same pattens as

$$\Pi x : nat\ (x = x)$$

But the red part *nat* must be of type *Type*

$Type : Type$? No way

Extra typing rules to form more products (polymorphism)

Polymorphism through rewriting

Express simple type theory as a first-order theory

$$\forall P\ (P \Rightarrow P)$$

$$\forall p\ (\varepsilon(p) \Rightarrow \varepsilon(p))$$

substitute by $ar(q,q)$

$$\varepsilon(ar(q,q)) \Rightarrow \varepsilon(ar(q,q))$$

Polymorphism through rewriting

Express simple type theory as a first-order theory

$$\forall P\ (P \Rightarrow P)$$

$$\forall p\ (\varepsilon(p) \Rightarrow \varepsilon(p))$$

substitute by $ar(q, q)$

$$\varepsilon(ar(q, q)) \Rightarrow \varepsilon(ar(q, q))$$

Need a rule

$$\varepsilon(ar(x, y)) \longrightarrow \varepsilon(x) \Rightarrow \varepsilon(y)$$

Proofs of simple type theory in $\lambda\Pi$ modulo

A uniform proof language

$\lambda\Pi$ modulo

A simple extension of $\lambda\Pi$ with a (parametric) congruence on types

Proofs of all axiom free theories in deduction modulo

Unifies many (more or less esoteric) type theories

Allows to design new type theories (for set theory, ...)

Uniformity allows uniform meta-theory (*e.g.* termination criteria)