The LogiCal team

Located on two sites

- INRIA-Rocquencourt (4 permanent + 3 PhD students)

- Orsay University and CNRS (4 permanent + 4 PhD students)

Both close to Paris

LogiCal: main activities

- Fundamental research on proof theory

- Development of the Coq proof assistant

- Proof development in Coq

The Coq proof assistant

Based on the Calculus of Inductive Constructions

- A logical strength comparable to ZF

- The functional expressivity of (a higher-order) ML

- The logical expressivity of natural deduction and inductively defined propositions

Not as automatized as PVS, ACL2, ... for proof-of-programs

Proof scripting not as mathematician-readable as Mizar

A Coq script

```
Fixpoint minus [n:nat] : nat -> nat :=
  [m:nat]Cases n m of
            O _ => O
         | (S k) O => (S k)
         | (S k) (S l) => (minus k l)
        end.

Lemma minus_plus_simpl :
        (n,m,p:nat)((minus n m)=(minus (plus p n) (plus p m))).
Proof.
  Induction p; Simpl; Auto with arith.
Qed.
```

A Coq script

```
Coq < Print minus_plus_simpl.
[n,m,p:nat]
 (nat_ind [n0:nat](minus n m)=(minus (plus n0 n) (plus n0 m))
   (refl_equal nat (minus n m))
   [p0:nat; IHp:((minus n m)=(minus (plus p0 n) (plus p0 m)))]IHp p)
     : (n,m,p:nat)(minus n m)=(minus (plus p n) (plus p m))
```

Rendered by HELM

```
DEFINITION minus_plus_simpl()
TYPE =
       for all n:nat.for all m:nat.for all p:nat.((n-m)=((p+n)-(p+m)))
BODY =
       lambda n:nat
         lambda m:nat
           lambda p:nat
             We prove ((n-m)=((p+n)-(p+m)))
             by induction on p
                 Case O
                     (refl_equal . .)
                      we proved  ((n-m)=(n-m))
                 Case (S n0:nat)
                     By induction hypothesis, we have:
                     (H) ((n-m)=((n0+n)-(n0+m)))
                     H
             we proved ((n-m)=((p+n)-(p+m)))
        we proved  for all n:nat. for all m:nat. for all p:nat. ((n-m)=((p+n)-(p+m)))
```

Main Coq libraries

- Arithmetic (Peano numbers and binary integers)

- Axiomatized real numbers (derivability, trigonometry, series, ...)

- Algebra (algebraic structures, fundamental theorem of algebra, categories, ...)

- Sets, relations, orders, logics

- Planar geometry

- Computer science structures (booleans, lists, maps, automatas, ...)

[13 standard libraries + 70 extra user contributions]

The Bologna HELM experiment

Export of Coq formal proofs to XML format

Application of XSLT filters for output in

- Natural language

- MathML Content

- MathML presentation

Computation of the dependency graph

Our problems

- Classification

- Retrieving

- Proof rendering

- Modularity

Classifying

- Thematically (but sometimes at the intersection of several fields)

- By author

- By version of Coq

Currently, classified by version, site, theory + thematic cross classification

Retrieving

- By pattern (but how to identify $n < p + 1$ and $n \leq p$)

- By isomorphism (e.g. $A \to \forall x.B(x) \to C \equiv \forall x.B(x) \wedge A \to C$)

- By name (but no standard: e.g. commutativity is sometimes called symmetry)

Proof rendering

- To be able to render a proof in human-understandable format (cf HELM and Natural experiments), but still problem with the detail levels

- Conflict between verbosity and automation

Modularity

- Currently no module

- To be able to express theories independently of a specific implementation

Relations with OMDoc?