Logic-Independent Proof Search in Logical Frameworks (short paper)

Michael Kohlhase[1] Florian Rabe[1] Claudio Sacerdoti Coen[2]
Jan Frederik Schaefer[1]

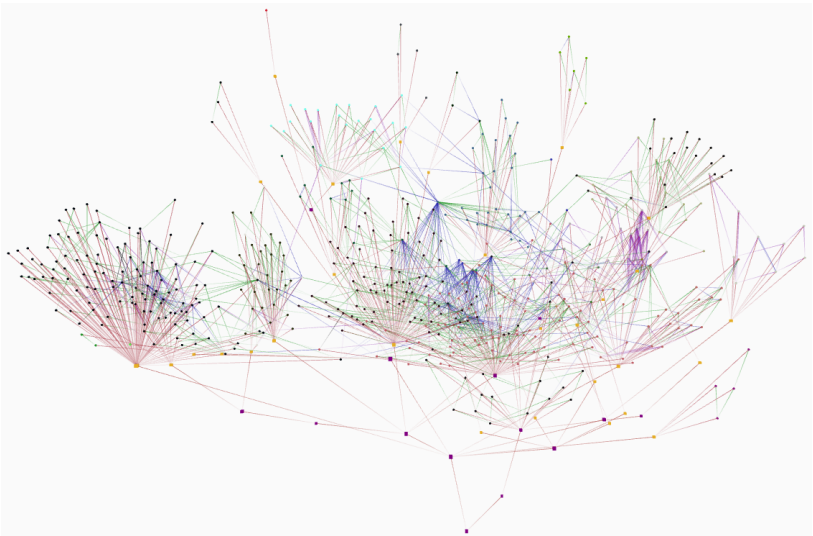
[1]FAU Erlangen-Nürnberg

[2]Università di Bologna

IJCAR
remotely from Erlangen, Germany
July 2, 2020

Logic development in MMT/LF

- Logical frameworks can be used to describe logics
- We have a tool for this: MMT *others exist*
  - → Large modular collection of logics *LATIN project*
- Free proof checking
- Wouldn't it be nice to also generate provers?
- We report on an experiment in this direction

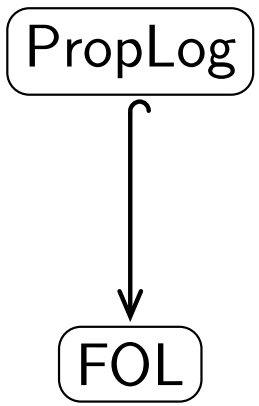


Logic Syntax in MMT/LF

theory PropLog =

prop : *type*

not : prop → prop
and : prop → prop → prop
or : prop → prop → prop
...



theory FOL =

include PropLog

term : *type*

forall : (term → prop) → prop *// Higher–order abstract syntax*
exists : (term → prop) → prop

Natural Deduction in MMT/LF

$$\frac{A \wedge B}{A}\ \text{andEl} \qquad \frac{A \vee B \qquad \begin{array}{c}[A]^1\\ \vdots\\ C\end{array} \qquad \begin{array}{c}[B]^1\\ \vdots\\ C\end{array}}{C}\ \text{orE}^1$$

theory PropLog_ND =
  include PropLog

  // ded X is type of proofs for X (judgments as types)
  ded : prop → *type*

  andEl : {A,B} ded (and A B) → ded A

  orE : {A,B,C} ded (or A B) → (ded A → ded C) →
                (ded B → ded C) → ded C

Generating Provers in ELPI

- ELPI is an extension of λProlog $\approx$ *Prolog + HOAS*
- Optimized for fast execution of logical algorithms

  type inference, unification, proof search, . . .

LF rule andEl : {A,B} ded (and A B) → ded A

ELPI equivalent

direct: **pi** *A* \ **pi** *B* \ ded (and *A B*) => ded *A*.

syn. sugar: ded *A* :– ded (and *A B*).

From LF to ELPI

Example: Or-Elimination

LF: orE : {A,B,C} ded (or A B) → (ded A → ded C) →
(ded B → ded C) → ded C

ELPI: ded *C* :– ded (or *A* *B*), ded *A* => ded *C*, ded *B* => ded *C*.

Example: Forall-Introduction

LF: forallI : {P} ({x} ded (P x)) → ded (forall P)

ELPI: ded (forall *P*) :– **pi** x \ ded (*P* x).

Controlling the Proof Search

- Problem: Search diverges *searching harder than checking*
- Solution: Control search with helper predicates:
  inspired by ProofCert project by Miller et al.
  - Intuition: Decide whether to apply rule
  - Do not affect correctness
  - Extra argument tracks aspects of proof state

Before: `ded A :-` `ded (and A B).`

Now: `ded X A :- help/andEl X A B X1, ded X1 (and A B).`

Helper Predicates

Name	**Predicate**	**Argument**
Iter. deepening	checks depth	remaining depth
Proof term	generates term	proof term
Product	calls other predicates	arguments for other predicates
Backchaining	Prolog's backchaining ($\approx$ forward reasoning from axioms via $\Rightarrow/\forall$ elimination rules)	pattern of formula to be proven (e.g. a conjunction)

Example helper: Iterative deepening

help/andEl (idcert *N*) _ _ (idcert *N1*) :– $N > 0$, *N1* **is** $N - 1$.

Tableau Provers

$$\frac{A \wedge B^F}{A^F \mid B^F}\ \text{andF} \qquad \frac{A \wedge B^F \qquad \begin{array}{c}[A^F] \\ \vdots \\ \bot\end{array} \qquad \begin{array}{c}[B^F] \\ \vdots \\ \bot\end{array}}{\bot}\ \text{andF}$$

LF: andF : {A,B} $A \wedge B^F \to (A^F \to \bot) \to (B^F \to \bot) \to \bot$

ELPI:
```
closed X :- help/andF X A B X1 X2 X3, f X1 (and A B),
            f/hyp A => closed X2, f/hyp B => closed X3.
```

With iterative deepening we get a working prover!
→ Other helpers result in more efficient provers

Conclusion

Summary:

- Develop logic in MMT/LF
- Generate ELPI provers from specified calculi
- Soundness w.r.t. MMT specification for free
- Can show soundness of calculus in MMT

Goals:

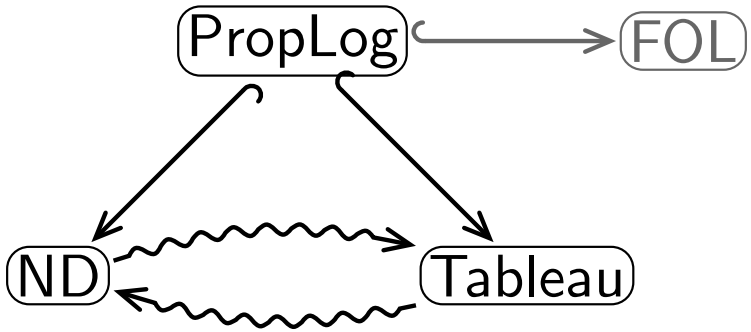
- Help logic developers with provers *MMT users*
- Help prover makers test strategies across logics *ELPI users*

Evaluation:

- Experiment at an early stage *more logics, strategies*
- Translation to FOL probably often more efficient
- Test run:

Generated (ND)	Me	Vampire
1.007 s (depth 9)	$\approx$ 16 min	0.001s

Bonus: Soundness and Completeness in MMT



- Express ND rules in terms of tableau rules
- Express tableau rules in terms of ND rules
- $\rightsquigarrow$ ND and tableau calculus can prove same set of propositions
- $\rightsquigarrow$ ND sound and complete $\Leftrightarrow$ tableau sound and complete

Bonus: Helper Predicates in ELPI (as they are generated)

```
% iterative deepening
help/andEl (idcert X3) A B (idcert X2) :- X3 > 0 , X2 is X3 - 1.

% record proof terms
help/andEl (ptcert (andEl A B X2)) A B (ptcert X2).

% combine helpers
help/andEl (prodcert X3/1 X3/2) A B (prodcert X2/1 X2/2) :-
    help/andEl X3/1 A B X2/1 , help/andEl X3/2 A B X2/2.

% back-chaining
help/andEl (bccert X3) A B (bccert (bc/fwdLocked X2)) :-
    bc/val X3 X4 , X4 > 0 , X2 is X4 - 1 , bc/fwdable (and A B).
```