Towards a broad spectrum proof certificate

Dale Miller

INRIA-Saclay & LIX, École Polytechnique
Palaiseau, France

MLPA, 15 July 2010

A talk full of speculations...

Outline

1. Dreaming
2. Sorting out some issues
3. Focused proof systems

Let's do some dreaming....

Wouldn't it be great to have *anti-gravity* devices?

- floating cities
- levitating cars and trains
- easy access to outer-space

Let's do some dreaming....

Wouldn't it be great to have *anti-gravity* devices?

- floating cities
- levitating cars and trains
- easy access to outer-space

Consider another dream, probably even more outlandish....

Let's do some dreaming....

Wouldn't it be great to have *anti-gravity* devices?

- floating cities
- levitating cars and trains
- easy access to outer-space

Consider another dream, probably even more outlandish....

Wouldn't it be great to have *proof certificates* that could be written by a broad range of deductive systems and easily validated by simple proof checkers.

Thus: any prover could trust a proof from any other prover.

Dream on: Need only ensure the correctness of the checker

The correctness of the prover is not a (central) issue.

There could be a *plethora of provers* in many styles:
specialized/general purpose, ad hoc/principled,
complete/incomplete, correct/buggy, etc.

We do not need to be unifying all theorem proving activity into one framework, such as Isabelle/X or Coq.

Dream on: A marketplace for proofs

The ACME company needs a formal proof for its next generation of controllers for airplanes, electric cars, medical equipment, etc.

ACME submit to the marketplace a proposed theorem: a proof certificate with a "hole" in it.

Anyone who can prove that theorem (or provide a counter-example) will get paid, provided that the alleged proof can be checked by ACME.

This marketplace could be wide open: anyone using any combination of deduction engines would be able to compete.

Dream on: Libraries for proofs

Once proofs have been written, they could be archived, searched, and retrieved.

Since proofs are supported by declarative techniques, they should have a life once checked and archived. One should be able to: transform them, browse them, apply them, etc.

One might *trust* the authority behind the library or not: you can check any retrieved proof yourself.

A library has strong motivations to be careful: accepting a non-proof puts their entire library and accumulative trust at risk. Such checking is in the public domain.

Which logic?

Classical or intuitionistic logic?

Imagine that these two logics fit together in one larger logic. Following Gentzen (LK/LJ), Girard (LU) and, recently, Liang & M.

First-order or higher-order?

Higher-order (à la Church 1940) seems a good choice since it includes propositional and first-order.

Modal, temporal, spatial?

I leave these out for now. One might incorporate modal operators directly; or encode their semantics; or use a "two-level logic" approach. There is likely to always be a frontier that does not fit...

Which proof system?

There are numerous, well studied proof systems: *natural deduction*, *sequent*, *tableaux*, *resolution*, etc.

Many others are clearly proof-like: *tables* (in model checking), *winning strategies* (in game playing), etc.

Other: *certificates for primality*, etc.

We wish to capture all of these proof objects.

- Pros: a deductive system does not need to transform its internal proof representation to something else entirely (a rather involved task).
- Cons: handling so many proof formats might make for a terribly complex proof checker.

Proofs will likely always be big

We need an effective execution model for proof checkers.

Allowing inference rules to closely model "actions" or "steps" in a given application domain should allow the engineer to provide natural proofs that are tailored to a given topic.

One can allow for a trade-off between *proof size* and *proof reconstruction*: a proof checker should be able to do some computation as well as some search.

A framework for addressing some of these issues

Focused proof systems provide a rich way to take the micro-rules of Gentzen's sequent calculus and to organize them into a wide range of macro-rules (all enjoying cut and initial elimination).

Linear logic can be used as a framework for the specification of inference rules. *Polarizing* such specifications differently allows one to simulate natural deduction, sequent calculus, tableaux, etc.

- Thus a simple interpreter for focused linear logic can be a proof checker for all of those proof systems.

Adding fixed points and equality allows one to build computation and model-checking into inference rules.

Focused proof systems

Invertible introduction inference rules:

$$\frac{\Gamma, B_1, B_2 \vdash \Delta}{\Gamma, B_1 \wedge B_2 \vdash \Delta} \qquad \frac{\Gamma \vdash \Delta, B[y/x]}{\Gamma \vdash \Delta, \forall x B}$$

The introduction rules for the de Morgan duals:

$$\frac{\Gamma, B[t/x] \vdash \Delta}{\Gamma, \forall x B \vdash \Delta} \qquad \frac{\Gamma_1 \vdash \Delta_1, B_1 \qquad \Gamma_2 \vdash \Delta_2, B_2}{\Gamma_1, \Gamma_2 \vdash \Delta_1, \Delta_2, B_1 \wedge B_2}$$

Focused proofs are built in two phases: one with only invertible rules (the "negative" or "asynchronous" phase); one with duals to the invertible rules (the "positive" or "synchronous" phase).

LKF : focused proof systems for classical logic

$$\frac{}{\vdash \Theta \Uparrow \Gamma, t^-} \qquad \frac{\vdash \Theta \Uparrow \Gamma, A \qquad \vdash \Theta \Uparrow \Gamma, B}{\vdash \Theta \Uparrow \Gamma, A \wedge^- B}$$

$$\frac{\vdash \Theta \Uparrow \Gamma}{\vdash \Theta \Uparrow \Gamma, f^-} \qquad \frac{\vdash \Theta \Uparrow \Gamma, A, B}{\vdash \Theta \Uparrow \Gamma, A \vee^- B} \qquad \frac{\vdash \Theta \Uparrow \Gamma, A[y/x]}{\vdash \Theta \Uparrow \Gamma, \forall x A}$$

LKF : focused proof systems for classical logic

$$\frac{}{\vdash \Theta \Uparrow \Gamma, t^-} \qquad \frac{\vdash \Theta \Uparrow \Gamma, A \qquad \vdash \Theta \Uparrow \Gamma, B}{\vdash \Theta \Uparrow \Gamma, A \wedge^- B}$$

$$\frac{\vdash \Theta \Uparrow \Gamma}{\vdash \Theta \Uparrow \Gamma, f^-} \qquad \frac{\vdash \Theta \Uparrow \Gamma, A, B}{\vdash \Theta \Uparrow \Gamma, A \vee^- B} \qquad \frac{\vdash \Theta \Uparrow \Gamma, A[y/x]}{\vdash \Theta \Uparrow \Gamma, \forall x A}$$

$$\frac{}{\vdash \Theta \Downarrow t^+} \qquad \frac{\vdash \Theta \Downarrow A \qquad \vdash \Theta \Downarrow B}{\vdash \Theta \Downarrow A \wedge^+ B} \qquad \frac{\vdash \Theta \Downarrow A_i}{\vdash \Theta \Downarrow A_1 \vee^+ A_2} \qquad \frac{\vdash \Theta \Downarrow A[t/x]}{\vdash \Theta \Downarrow \exists x A}$$

LKF : focused proof systems for classical logic

$$\frac{}{\vdash \Theta \Uparrow \Gamma, t^-} \qquad \frac{\vdash \Theta \Uparrow \Gamma, A \qquad \vdash \Theta \Uparrow \Gamma, B}{\vdash \Theta \Uparrow \Gamma, A \wedge^- B}$$

$$\frac{\vdash \Theta \Uparrow \Gamma}{\vdash \Theta \Uparrow \Gamma, f^-} \qquad \frac{\vdash \Theta \Uparrow \Gamma, A, B}{\vdash \Theta \Uparrow \Gamma, A \vee^- B} \qquad \frac{\vdash \Theta \Uparrow \Gamma, A[y/x]}{\vdash \Theta \Uparrow \Gamma, \forall x A}$$

$$\frac{}{\vdash \Theta \Downarrow t^+} \qquad \frac{\vdash \Theta \Downarrow A \qquad \vdash \Theta \Downarrow B}{\vdash \Theta \Downarrow A \wedge^+ B} \qquad \frac{\vdash \Theta \Downarrow A_i}{\vdash \Theta \Downarrow A_1 \vee^+ A_2} \qquad \frac{\vdash \Theta \Downarrow A[t/x]}{\vdash \Theta \Downarrow \exists x A}$$

Init

$$\frac{}{\vdash \neg P, \Theta \Downarrow P}$$

Store

$$\frac{\vdash \Theta, C \Uparrow \Gamma}{\vdash \Theta \Uparrow \Gamma, C}$$

Release

$$\frac{\vdash \Theta \Uparrow N}{\vdash \Theta \Downarrow N}$$

Decide

$$\frac{\vdash P, \Theta \Downarrow P}{\vdash P, \Theta \Uparrow \cdot}$$

P positive; N negative; C positive or negative literal

An example

Assume that Θ contains the formula $a \wedge^+ b \wedge^+ \neg c$ and that we have a derivation that Decides on this formula.

$$\dfrac{\dfrac{}{\vdash \Theta \Downarrow a}\ \mathit{Init} \quad \dfrac{}{\vdash \Theta \Downarrow b}\ \mathit{Init} \quad \dfrac{\dfrac{\vdash \Theta, \neg c \Uparrow \cdot}{\vdash \Theta \Uparrow \neg c}}{\vdash \Theta \Downarrow \neg c}\ \mathit{Release}}{\dfrac{\vdash \Theta \Downarrow a \wedge^+ b \wedge^+ \neg c}{\vdash \Theta \Uparrow \cdot}\ \mathit{Decide}}\ \mathit{and}$$

This derivation is possible iff Θ is of the form $\neg a, \neg b, \Theta'$. Thus, the "macro-rule" is

$$\dfrac{\vdash \neg a, \neg b, \neg c, \Theta' \Uparrow \cdot}{\vdash \neg a, \neg b, \Theta' \Uparrow \cdot}$$

Two certificates for propositional logic: negative

Use $\wedge^-$ and $\vee^-$. Their introduction rules are invertible. The initial "macro-rule" is huge, having all the clauses in the conjunctive normal form of B as premises.

$$\dfrac{\dfrac{\dfrac{}{\vdash L_1, \ldots, L_n \Downarrow L_i}\,\mathit{Init}}{\vdash L_1, \ldots, L_n \Uparrow \cdot}\,\mathit{Decide} \quad \ldots}{\vdots}\ \ldots$$

$$\dfrac{}{\vdash \cdot \Uparrow B}$$

The proof certificate can specify the complementary literals for each premise or it can ask the checker to *search* for them.

Proof certificates can be tiny but require exponential time for checking.

Two certificates for propositional logic: positive

Use $\wedge^+$ and $\vee^+$. Sequents are of the form $\vdash B, \mathcal{L} \Uparrow \cdot$ and $\vdash B, \mathcal{L} \Downarrow P$, where B is the original formula to prove, P is positive, and $\mathcal{L}$ is a set of negative literals.

Macro rules are in one-to-one correspondence with $\phi \in DNF(B)$. Divide ϕ into ϕ^- (negative literals) and ϕ^+ (positive literals).

$$\frac{\frac{\{\vdash B, \mathcal{L}, N \Uparrow \cdot \quad | \quad N \in \phi^-\}}{\vdash B, \mathcal{L} \Downarrow B}\ \textit{Decide}}{\vdash B, \mathcal{L} \Uparrow \cdot} \text{ provided } \neg\phi^+ \in \mathcal{L}$$

Proof certificates are sequences of members of $DNF(B)$. Size and processing time can be reduced (in response to "cleverness").

Equality and Fixed Points as connectives

$$\frac{}{\vdash \Theta \Downarrow t = t} \qquad \frac{}{\vdash \Theta \Uparrow \Gamma, s \neq t}\ \ddagger \qquad \frac{\vdash \Theta\sigma \Uparrow \Gamma\sigma}{\vdash \Theta \Uparrow \Gamma, s \neq t}\ \dagger$$

$\ddagger$ s and t are not unifiable.

$\dagger$ s and t to be unifiable and σ to be their mgu

Equality and Fixed Points as connectives

$$\frac{}{\vdash \Theta \Downarrow t = t} \qquad \frac{}{\vdash \Theta \Uparrow \Gamma, s \neq t}\ \ddagger \qquad \frac{\vdash \Theta\sigma \Uparrow \Gamma\sigma}{\vdash \Theta \Uparrow \Gamma, s \neq t}\ \dagger$$

$\ddagger$ s and t are not unifiable.

$\dagger$ s and t to be unifiable and σ to be their mgu

$$\frac{\vdash \Theta \Uparrow \Gamma, B(\mu B)\bar{t}}{\vdash \Theta \Uparrow \Gamma, \mu B\bar{t}} \qquad \frac{\vdash \Theta \Downarrow B(\mu B)\bar{t}}{\vdash \Theta \Downarrow \mu B\bar{t}}$$

B is a formula with $n \geq 0$ variables abstracted; $\bar{t}$ is a list of n terms.

Here, μ denotes neither the least nor the greatest fixed point. That distinction arises if we add induction and co-induction.

Examples of fixed points

Natural numbers: terms over 0 for zero and s for successor. Two ways to define predicates over numbers.

$$\begin{array}{rcl} \mathit{nat}\ 0 & :- & \mathit{true}. \\ \mathit{nat}\ (s\ X) & :- & \mathit{nat}\ X. \\ \mathit{leq}\ 0\ Y & :- & \mathit{true}. \\ \mathit{leq}\ (s\ X)\ (s\ Y) & :- & \mathit{leq}\ X\ Y. \end{array}$$

Above, as a logic program and below, as fixed points.

$$\mathit{nat} = \mu(\lambda p \lambda x.(x = 0) \vee^{+} \exists y.(s\ y) = x \wedge^{+} p\ y)$$

$$\mathit{leq} = \mu(\lambda q \lambda x \lambda y.(x = 0) \vee^{+} \exists u \exists v.(s\ u) = x \wedge^{+} (s\ v) = y \wedge^{+} q\ u\ v).$$

Horn clauses can be made into fixed point specifications (mutual recursions requires standard encoding techniques).

The engineering of proof systems

Consider proving the positive focused sequent

$$\vdash \Theta \Downarrow (\mathit{leq}\ m\ n \wedge^{+} N_1) \vee^{+} (\mathit{leq}\ n\ m \wedge^{+} N_2),$$

where m, n are natural numbers and N_1, N_2 are negative formulas. There are exactly two possible macro rules:

$$\frac{\vdash \Theta \Downarrow N_1}{\vdash \Theta \Downarrow (\mathit{leq}\ m\ n \wedge^{+} N_1) \vee^{+} (\mathit{leq}\ n\ m \wedge^{+} N_2)} \text{ for } m \leq n$$

$$\frac{\vdash \Theta \Downarrow N_2}{\vdash \Theta \Downarrow (\mathit{leq}\ m\ n \wedge^{+} N_1) \vee^{+} (\mathit{leq}\ n\ m \wedge^{+} N_2)} \text{ for } n \leq m$$

A macro inference rule can contain an entire Prolog-style computation.

The engineering of proof systems (cont)

Consider proofs involving simulation.

$$sim\ P\ Q\ \equiv\ \forall P' \forall A[\ P \xrightarrow{A} P' \supset \exists Q'\ [Q \xrightarrow{A} Q' \wedge sim\ P'\ Q']].$$

Typically, $P \xrightarrow{A} P'$ is given as a table or as a recursion on syntax (*e.g.*, CCS): hence, as a fixed point.
The body of this expression is exactly two "macro connectives".

- $\forall P' \forall A[P \xrightarrow{A} P' \supset \cdot\ ]$ is a negative "macro connective". There are no choices in expanding this macro rule.
- $\exists Q'[Q \xrightarrow{A} Q' \wedge^{+} \cdot\ ]$ is a positive "macro connective". There can be choices for continuation Q'.

These macro-rules now match exactly the sense of simulation.

Some references

[1] Jean-Marc Andreoli. Logic programming with focusing proofs in linear logic. *J. of Logic and Computation*, 2(3):297–347, 1992.

[2] Chuck Liang and Dale Miller. Focusing and polarization in linear, intuitionistic, and classical logics. *Theoretical Computer Science*, 410(46):4747–4768, 2009.

[3] Dale Miller and Vivek Nigam. Incorporating tables into proofs. In J. Duparc and T. A. Henzinger, editors, *CSL 2007: Computer Science Logic*, volume 4646 of *LNCS*, pages 466–480. Springer, 2007.

[4] Vivek Nigam and Dale Miller. Focusing in linear meta-logic. In *Proceedings of IJCAR: International Joint Conference on Automated Reasoning*, volume 5195 of *LNAI*, pages 507–522. Springer, 2008.

The papers with Miller as a co-author are on his web page.