Towards a broad spectrum proof certificate

Dale Miller

INRIA-Saclay & LIX, École Polytechnique
Palaiseau, France

Carnegie Mellon University, 17 October 2011

Can we standardize, communicate, and trust formal proofs?

Outline

We must first narrow our topic

Proofs are *documents* that are used to *communicate trust* within a *community of agents*.

Agents can be machines and humans.

Our focus:
publishing and checking formal proofs by computer agents

Not our focus today: learning from proofs, interacting with proofs, compute with proofs.

Provers: computer agents that produce proofs

There is a wide range of provers.

- automated and interactive theorem provers
- model checkers, SAT solvers
- type inference, static analysis
- testers

There is a wide range of "evidence" of proof.

- proof scripts: steer a theorem prover to a proof
- resolution refutations, natural deduction, tableaux, etc
- winning strategies, simulations

It is the exception when one prover's evidence is shared with another prover.

Require provers to publish their proofs

Since provers do not currently communicate proofs, the trend is to unifying various theorem proving activities into existing frameworks, eg, Isabelle or Coq.

Separate proofs from provenance: insist that provers output their proofs so others can check them.

We shall use the term “proof certificate” for those documents denoting proofs that are circulated between provers.

Outline

D1: A simple checker can, in principle, check if a proof certificate denotes a proof.

The *de Bruijn's principle:* provers should output proofs that can be checked by *simple* checkers. Here "simple" might mean that the checker can be independently validated (eg, by hand).

Ultimately, I will argue that proof certificates will be programs and a checker will be an interpreter for such programs.

"Everything should be made as simple as possible,
but not one bit simpler."
-Albert Einstein

D2: The proof certificate format supports a broad spectrum of proof systems.

One should not need to radically transform accumulated proof evidence in order to output a proof certificate.

Clearly, there is a tension between **D1** and **D2**.

Consider the following additional consequences of these two desiderata.

Marketplaces for proofs

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

ACME submits to the “proofs” marketplace a proposed theorem as a proof certificate with a “hole” for its actual proof.

The contract: You get paid if you can fill the hole in such a way that ACME can check it.

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

Marketplaces for proofs

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

ACME submits to the "proofs" marketplace a proposed theorem as a proof certificate with a "hole" for its actual proof.

The contract: You get paid if you can fill the hole in such a way that ACME can check it.

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

Providing a *partial proof* or a *counter-example* should also have some economic value: these should be allowed in a more general setting of "proof certificates".

Libraries of proofs

Proof certificates can be archived, searched, and retrieved.

Additionally, one might be able to browse, apply, and transform them.

One might *trust* the authority behind the library.

Libraries might invest in significant computing power, thus expanding the proof certificates that they can check.

A library has strong motivations to be careful: accepting a non-proof puts their entire library and accumulative trust at risk.

D3: A proof certificate is intended to denote a proof in the sense of structural proof theory.

Structural proof theory is a mature field that deals with deep aspects of proofs and their properties.

For example: given certificates for

$$\forall x(A(x) \supset \exists y \ B(x,y)) \quad \text{and} \quad A(10),$$

can we extract from them a witness t such that $B(10,t)$ holds?

D4: A proof certificate can simply leave out details of the intended proof.

Formal proofs are often huge. All means to reduce their size need to be available.

- Introductions of abstractions and lemma.
- Separate *computation* from *deduction* and leave computation traces out of the certificate.
- Allow trade-offs between *proof size* and *proof reconstruction*: (bounded) proof search maybe need to fill in holes.

This desideratum leads to strong demands on the nature of proof certificates.

- What bound on search is sensible?
- How to ensure that such search is sensibly directed?

Outline

Which logic?

First-order or higher-order?

Which logic?

First-order or higher-order? Both!

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

Which logic?

First-order or higher-order? Both!

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

Classical or intuitionistic logic?

Which logic?

First-order or higher-order? Both!

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

Classical or intuitionistic logic? Both!

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

Which logic?

First-order or higher-order? Both!

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

Classical or intuitionistic logic? Both!

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

Modal, temporal, spatial?

I leave these out for now. There is likely to always be a frontier that does not fit. (However, the syntax of modal operators fits well with Church's logic and their semantics can similarly be encoded.)

Which computation paradigm?

Proof certificates will need to be “performed” and reconstructed.

Checking can be computationally expensive.

Since our view of computation should be broad spectrum as well, it should be

- non-deterministic, since determinism is a special case;
- concurrent, since sequential is a special case; and
- relational, since functions are a special case.

Logic programming might be a good candidate.

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.

Of course, handling so many proof formats might make for a terribly complex proof checker.

Atoms and molecules of inference

We outline how all these demands on certificates can be addressed using what we know of the theory of proof structures.

There are **atoms of inference**.

- Gentzen's **sequent calculus** first provided these: introduction and structural rules.
- Girard's **linear logic** refined our understanding of these further.
- To account for first-order structure, we also need **fixed points** and **equality**.

There are **molecules of inference**.

- There are "rules of chemistry" for assembling atoms of inference into molecules of inference ("synthetic inference rules").

Satisfying the desiderata

D1: Simple checkers.
Only the atoms of inference and the rules of chemistry (both small and closed sets) need to be implemented in the checker.

D2: Certificates supports a wide range of proof systems.
The molecules of inference can be engineered into a wide range of existing inference rules.

D3: Certificates are based on proof theory.
Immediate by design.

D4: Details can be elided.
Proof search in the space of atoms can match proof search in the space of molecules. (Don't want to invent new molecules in the checker.)

Outline

Focused proof systems

Consider a one-side sequent calculus system for classical logic.

Two *invertible* introduction inference rules:

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

The inference rules for their de Morgan duals (not invertible):

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

Focused proofs are built in *two phases*:

- the “up arrow” $\Uparrow$ phase where one only has invertible rules
- the “down arrow” $\Downarrow$ phase where one has (not-necessarily) invertible rules

LKF : (multi)focused proof systems for classical logic

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

$$\frac{}{\vdash \Theta \Downarrow t^+} \qquad \frac{\vdash \Theta \Downarrow \Gamma_1, B_1 \qquad \vdash \Theta \Downarrow \Gamma_2, B_2}{\vdash \Theta \Downarrow \Gamma_1, \Gamma_2, B_1 \wedge^+ B_2} \qquad \frac{\vdash \Theta \Downarrow \Gamma, B_i}{\vdash \Theta \Downarrow \Gamma, B_1 \vee^+ B_2}$$

Init Store Release Decide

$$\frac{}{\vdash \neg P_a, \Theta \Downarrow P_a} \qquad \frac{\vdash \Theta, C \Uparrow \Gamma}{\vdash \Theta \Uparrow \Gamma, C} \qquad \frac{\vdash \Theta \Uparrow \mathcal{N}}{\vdash \Theta \Downarrow \mathcal{N}} \qquad \frac{\vdash \mathcal{P}, \Theta \Downarrow \mathcal{P}}{\vdash \mathcal{P}, \Theta \Uparrow \cdot}$$

$\mathcal{P}$ multiset of positives; $\mathcal{N}$ multiset of negatives;
P_a positive literal; C positive formula or negative literal

Results about LKF

Let B be a propositional logic formula and let $\hat{B}$ result from B by placing $+$ or $-$ on t, f, $\wedge$, and $\vee$ (there are exponentially many such placements).

Theorem. B is a tautology if and only if $\hat{B}$ has an LKF proof. [Liang & M, TCS 2009]

Thus the different polarizations do not change *provability* but can radically change the *proofs*.

Also:

- Negative (non-atomic) formulas are treated linearly (never weakened nor contracted).
- Only positive formulas are contracted (in the Decide rule).

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.

$$
\dfrac{
  \dfrac{}{\vdash \Theta \Downarrow a}\ \mathit{Init}
  \qquad
  \dfrac{}{\vdash \Theta \Downarrow b}\ \mathit{Init}
  \qquad
  \dfrac{\dfrac{\vdash \Theta, \neg c \Uparrow \cdot}{\vdash \Theta \Uparrow \neg c}}{\vdash \Theta \Downarrow \neg c}\ \begin{array}{l}\mathit{Release}\\ \mathit{and}\end{array}
}{
  \dfrac{\vdash \Theta \Downarrow a \wedge^+ b \wedge^+ \neg c}{\vdash \Theta \Uparrow \cdot}\ \mathit{Decide}
}
$$

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

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

A certificates for propositional logic: compute CNF

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.

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

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.

Positive connectives allow for inserting information

Let B have several alternations of conjunction and disjunction.

Using positive polarities with the tautology $C = (p \vee^{+} B) \vee^{+} \neg p$ allows for a more clever proof then the previous one.

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash C, \neg p \Downarrow p}}{\vdash C, \neg p \Downarrow C}\;{*}}{\vdash C, \neg p \Uparrow \cdot}\;\mathit{Decide}}{\vdash C \Uparrow \neg p}}{\vdash C \Downarrow \neg p}}{\vdash C \Downarrow C}\;{*}}{\vdash C \Uparrow \cdot}\;\mathit{Decide}}{\vdash \cdot \Uparrow C}
$$

Clever choices $*$ are injected twice. The subformula B is avoided.

First-order terms and their structure

$$\frac{\vdash \Theta \Uparrow \Gamma, A[y/x]}{\vdash \Theta \Uparrow \Gamma, \forall x\ A}\ \S \qquad \frac{\vdash \Theta \Downarrow \Gamma, A[t/x]}{\vdash \Theta \Downarrow \Gamma, \exists x\ A}$$

§ y is not free in the lower sequent

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

‡ s and t are not unifiable. † s and t have mgu σ.

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

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

Here, μ and ν denotes some fixed point. Least and greatest require 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.

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

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 down-arrow focused sequent

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

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

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

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

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 from model theory / concurrency theory.

Example: Resolution as a proof certificate

A *clause:* $\forall x_1 \ldots \forall x_n [L_1 \vee \cdots \vee L_m]$
A *negated clause:* $\exists x_1 \ldots \exists x_n [L_1 \wedge \cdots \wedge L_m]$

1. A clause C is *trivial* if it contains complementary literals.
2. A clause C_1 *subsumes* C_2 if there is a substitution instance of the literals in C_1 which is a subset of the literals in C_2.
3. C_3 is a *resolution* of C_1 and C_2 if we chose the mgu of two complementary literals, one from each of C_1 and C_2, etc.

Polarize using $\vee^-$ and $\wedge^+$.
Let $\vdash_d \Theta \Uparrow \Gamma$ mean that $\vdash \Theta \Uparrow \Gamma$ has a proof with decide depth d.

- If C is trivial then $\vdash_1 \cdot \Uparrow C$.
- If C_1 subsumes a non-trivial clause C_2 then $\vdash_1 \neg C_1 \Uparrow C_2$.
- If C_3 is a resolvent of C_1 and C_2 then $\vdash_2 \neg C_1, \neg C_2 \Uparrow C_3$.

Example: Resolution as a proof certificate (cont)

Translate a refutation of $C_1, \ldots, C_n$ into an LKF proof with small holes as follows:

$$\dfrac{\begin{array}{c}\Xi\\ \vdash \neg C_1, \neg C_2 \Uparrow C_{n+1}\end{array} \qquad \dfrac{\begin{array}{c}\vdots\\ \vdash \neg C_1, \ldots, \neg C_n, \neg C_{n+1} \Uparrow \cdot\end{array}}{\vdash \neg C_1, \ldots, \neg C_n \Uparrow \neg C_{n+1}}\ \mathit{Store}}{\vdash \neg C_1, \ldots, \neg C_n \Uparrow \cdot}\ \mathit{Cut}_p$$

Here, Ξ can be replaced with a "hole" annotated with bound 2.

To capture more of the parallel structure present in a refutation, a "multicut" can be used here.

The λ-cube, $\lambda\Pi$, and deduction modulo

Functional Pure Type Systems can be embedded in the $\lambda\Pi$-calculus modulo. [Cousineau & Dowek; TLCA 2007]

- Functional computations (eg, β-reductions) are performed by the “modulo” aspect.
- The Dedukti proof checker [Boespflug; phd 2011] implements $\lambda\Pi$-calculus modulo by compilation into Haskell.

$\lambda\Pi$-calculus (LF) can be embedded into intuitionistic logic (λProlog) [Snow, Baelde, Nadathur; PPDP 10].

- Functional computations can be captured by logic programs.

Can type theory be meaningfully seen as a molecular construction on top of intuitionistic logic?

ProofCert: Broad Spectrum Proof Certificates

My application for an ERC Advanced Investigator Grant ("high risk, high gain") on this topic has been recently accepted.

Budget details to be worked out but the key points should be:

- five years duration (2012 - 2016)
- 2.2 million euros total.
- three PhD grants (each lasting 3 years)
- eight years of PostDoc support
- multiyear funding for an engineer

Interested in living in and working near Paris?

Some recent references

[1] Liang and Miller. Focusing and polarization in linear, intuitionistic, and classical logics. *Theoretical Computer Science*, 2009.

[2] Liang and Miller. *Kripke Semantics and Proof Systems for Combining Intuitionistic Logic and Classical Logic*, 2011 (submitted).

[3] Nigam and Miller. A framework for proof systems, *J. of Automated Reasoning*, 2010.

[4] Miller, Communicating and trusting proofs: The case for broad spectrum proof certificates, draft available on web.

[5] Miller, A proposal for broad spectrum proof certificates, *CCP 2011: Certified Proofs and Programs*.