ProofCert: Broad Spectrum Proof Certificates

Dale Miller

INRIA-Saclay & LIX, École Polytechnique
Palaiseau, France

INRIA's Scientific Council, 18 November 2011

an Advanced Investigator Grant ("High risk, high gain")
accepted by the
Seventh Framework Programme
"Ideas" specific programme
European Research Council

We must first narrow our topic

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

- In general, agents can be machines or humans.

- Our focus: publishing and checking *formal proofs* by computer *agents*

- Not our focus (yet): reading and learning from proofs, interacting with proofs, computing 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 that 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.

Goal: A sea change is needed in formal methods

Sun Microsystems (1984): The network *is* the computer

The formal methods community uses many isolated provers technologies: proof assistants (Coq, Isabelle, HOL, PVS, etc), model checkers, SAT solvers, etc.

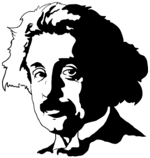
Goal: Permit the formal methods community to become a network of communicating and trusting provers.

We shall use the term "proof certificate" for those documents denoting proofs that are circulated and checked.

Four desiderata for proof certificates

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).



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

Almost certainly, proof certificates will themselves be programs and a checker will be an interpreter for such programs.

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

One should not need to radically transform your system's proof evidence in order to output a proof certificate.

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

Consider the following 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. The general setting of "proof certificates" should allow for these.

Libraries of proofs

Proof certificates can be archived, searched, and retrieved.

One should be able to browse, apply, and transform them.

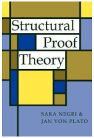
One might *trust* the authority behind the library.

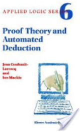
Libraries can invest in significant computing power, thus expanding the proof certificates that they can check.

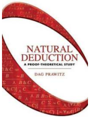
A library has strong motivations to be careful: accepting a non-proof puts their entire library and accumulative trust at risk.

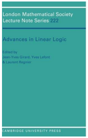
D3: A proof certificate is intended to denote a proof in the sense of structural proof theory.

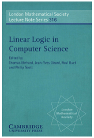
Structural proof theory is a mature field that deals with deep aspects of proofs and their properties.









For example: given certificates for $\forall x(A(x) \supset \exists y\ B(x,y))$ and $A(10)$, can we extract from them a t such that $B(10,t)$ holds?

Such proofs can also be considered **immortal**.

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 (cut introductions).
- 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.

D4 leads to challenging demands on proof certificates.

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

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!

There are efforts to put these two logics together in one larger logic: 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!

There are efforts to put these two logics together in one larger logic: Gentzen (LK/LJ), Girard (LU) and, recently, Liang & M.

Modal, temporal, spatial?

Leave these out for now: there is likely to always be a frontier that does not fit. (However, the syntax and semantics of many modal operators fit well with Church's logic.)

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.

How can a proof checker for so many formats be "simple?"

Atoms and molecules of inference

About seven years of *basic research* into proof theory suggests that all these desiderata can be based on the following principles.

There are **atoms of inference**.

- Gentzen's **sequent calculus** first uncovered these: introduction and structural rules.
- Girard's **linear logic** refined our understanding of these further.
- **Fixed points** and **equality** account for first-order structures.

There is a **chemistry** that provides rules for assembling atoms into molecules of inference (following *focused proof systems*).

One can build such **molecules of inference** to match a great range of proof structures.

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. (Computation can be placed inside 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. (The checker does not invent new molecules.)

Resources provided and committed

Budget negotiations are now completed.

Signatures are all that remain.

- five years duration (2012 - 2016)
- 2.2 million euros
- three PhD grants (each lasting 3 years)
- eight years of PostDoc support
- multiyear funding for an engineer
- funds for interns, short-term, long-term visitors
- 70% of the PI's time