**Proofs and Justification**

Konstantine Arkoudas
konstantine@alum.mit.edu
http://www.cag.lcs.mit.edu/~kostas/dpls/athena

Selmer Bringsjord
selmer@rpi.edu
http://www.rpi.edu/~brings

Dept of Cognitive Science
Dept of Computer Science
Rensselaer Polytechnic Institute (RPI)
Troy NY 12180 USA

ECAP 06 @ NTNU 6.23.06

# Computer Proofs and Justification

Konstantine Arkoudas
`konstantine@alum.mit.edu`
http://www.cag.lcs.mit.edu/∼kostas/dpls/athena

Selmer Bringsjord
`selmer@rpi.edu`
http://www.rpi.edu/∼brings

Dept of Cognitive Science
Dept of Computer Science
Rensselaer Polytechnic Institute (RPI)
Troy NY 12180 USA

ECAP 06 @ NTNU 6.23.06

**Computer Proofs and Justification**

**(On Foxes and Hedgehogs)**

**(On Engineering-Guided Philosophy)**

Konstantine Arkoudas
`konstantine@alum.mit.edu`
http://www.cag.lcs.mit.edu/∼kostas/dpls/athena

Selmer Bringsjord
`selmer@rpi.edu`
http://www.rpi.edu/∼brings

Dept of Cognitive Science
Dept of Computer Science
Rensselaer Polytechnic Institute (RPI)
Troy NY 12180 USA

ECAP 06 @ NTNU 6.23.06

**The Four-Color Theorem (4CT)**

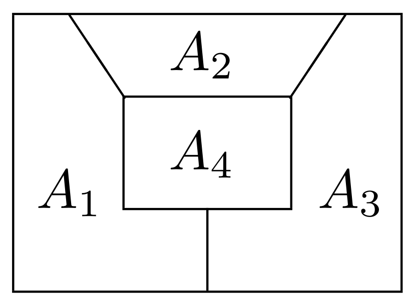
Using no more than 4 colors, every map can be colored so that adjacent countries always have distinct colors.

First formulated around 1840-1850 (Moebius, DeMorgan).

Kempe published a buggy proof in 1879.

Heawood found the error in 1890, and proved that 5 colors suffice.

It's clear that *at least* 4 colors are necessary:



## The Appel-Haken Proof

In 1976, Appel and Haken proved the conjecture for four colors.

Their proof, at some point, had to perform a very large case analysis that was not feasible by hand.

They wrote specialized computer code for it.

The analysis required a lot of computing power (for those days): 1200 hours on 4 computers.

## Philosophical Reaction

Shortly after the A+H proof, Tymoczko claimed that *if* we accept 4CT as a theorem, *then*:

1. We are changing the *concept of mathematical proof*.
2. Mathematics becomes much more like an experimental natural science.
3. In particular, deduction ceases to be the chief methodology of mathematics.

It would also follow that:

- the concept of proof is *negotiable*, and
- standards of rigor are not immutable.

## Rationale

Premise: "Proofs are *surveyable*."

A $\text{proof}_t$ must be such that mathematicians can look it over, review it, and verify it.

But no mathematician has ever surveyed a $\text{proof}_t$ of the 4CT.

Indeed, most probably no mathematician ever will.

Therefore, the A+H experiment is not a $\text{proof}_t$ of 4CT.

## Social Constructivism

Tymoczko's conclusions are aligned with social constructivism in mathematics:

1. Mathematics is an intrinsically *human* activity.
2. The main vehicle for generating mathematical knowledge is *not deduction*.
3. Mathematical truth is *not necessary*.
4. Mathematical knowledge is *not a priori*.
5. Mathematical knowledge is *not infallible*.
6. Mathematical rigor is a *changing* social construction.

## Orthodox Reaction

Computer methods offer:

1. not a new *concept* of proof, but rather
2. a new way of *discovering*, *presenting* and *checking* proofs.

Likewise, what is negotiable is:

1. not the underlying concept of proof, but
2. our techniques for *checking* whether an object really represents a correct proof.

## The *A Priori* Question

Tymoczko said that the evidence one obtains from an unsurveyable computer proof depends on empirical factors (reliability of computers, etc.)

Hence, the corresponding justification is not *a priori*.

But the evidence one obtains from *most* proofs (even surveyable, non-computerized ones) depends on empirical factors.

Every time a physical agent $A$ evaluates a proof, empirical considerations become causally relevant.

Whether it is silicon or human brains, any physical mechanism is subject to error. Hidden appeals to induction are often made.

## More on the *A Priori* Question

Mathematician $A$ checks a *token* $\widehat{\pi}$ of a proof $\pi$.

$A$ *thinks* the proof is sound, but is not sure.

$A$ thinks he knows how to apply *modus ponens*, having done it with apparent success many times before.

But $A$ has the flu.

$A$ checks $\widehat{\pi}$ again (repeating the "experiment").

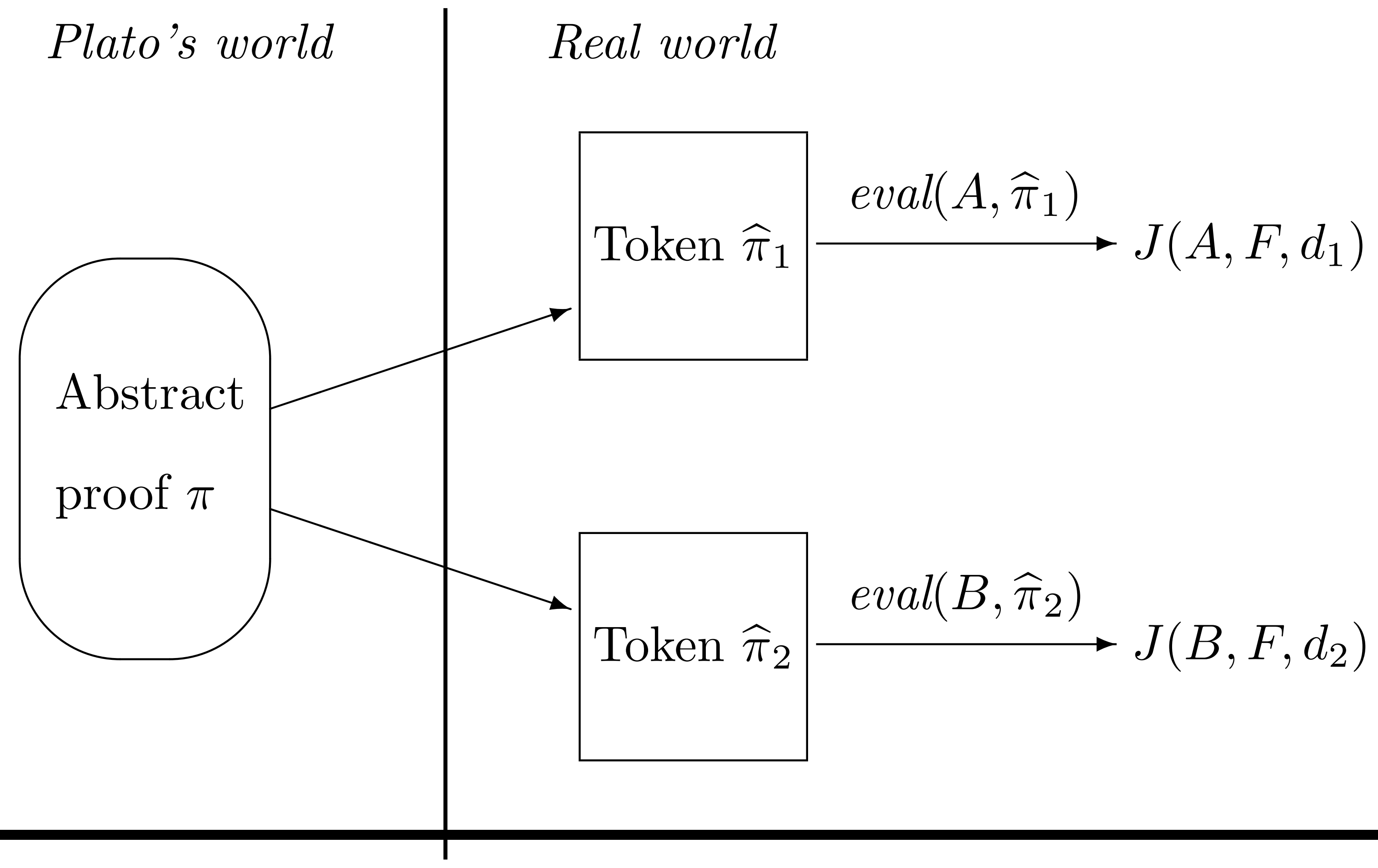
$A$ comes across a typo. The token $\widehat{\pi}$ does not instantiate $\pi$ correctly after all.

Then $A$ asks $B$ and $C$ to also check $\widehat{\pi}$ (for *redundancy*).

Are these "experimental" or "inductive" techniques?

## The Big Picture

$J(A, F, d)$ : Person $A$ is justified in believing $F$ to degree $d \in [0, 1]$



## What Tymoczko Missed

He viewed computers as black boxes.

Analogous to Martian inference rule “Simon says.”

But computers are *not* black boxes.

We have detailed mathematical theories that *explain* and *predict* their observable behavior.

In particular, we can prove *theorems* about computer programs, such as:

$$\forall\, x, y\ .\ [\mathcal{P}(x) \hookrightarrow y] \Rightarrow R(x, y)$$

I.e.: If and when program $\mathcal{P}$ produces the output $y$ given input $x$, then $R(x, y)$. Moreover, *those* proofs can be surveyable — if $P$ is small/simple. (E.g., ‘R’ might denote correctness.)

## Believing in Computer-Generated Results

But if we believe

$$\forall\, x, y \;.\; [\mathcal{P}(x) \hookrightarrow y] \Rightarrow R(x, y) \tag{1}$$

and we also believe

$$[\mathcal{P}(a) \hookrightarrow b] \tag{2}$$

then we are entitled to believe

$$R(a, b).$$

The theoretical question is: What justification, in general, can we have for believing (1) and (2)?

The practical question is: Is it possible to engineer systems in a way that maximizes such justification?

## Forming Such Beliefs

$$\boxed{\text{START}}$$

Stage 1 (mosty deductive)

$$\big\downarrow \; eval(A, \widehat{\pi}) \text{ (token proof of correctness } \mathcal{P}\text{)}$$

$$J(A, \forall\, x, y \;.\; [\mathcal{P}(x) \hookrightarrow y] \Rightarrow R(x, y), d_1)$$

Stage 2 (mosty empirical)

$$\big\downarrow \; observe(A, \widehat{\mathcal{P}}(\widehat{a}) \hookrightarrow \widehat{b})$$

$$J(A, \mathcal{P}(a) \hookrightarrow b, d_2)$$

Stage 3 (deductive)

$$\big\downarrow \; infer(A, R(a, b))$$

$$J(A, R(a, b), d_3)$$

What does $d_1$ depend on? How about $d_2$?

## Analyzing the Justification Degrees

$d_1$ depends on:

[1] The size and logical complexity of $\widehat{\pi}$

$d_2$ depends on:

[2] The size of $\widehat{\mathcal{P}}$

[3] The size of $\widehat{a}$

[4] The size of $\widehat{b}$

[5] The length of the computation of $\widehat{P}(\widehat{a})$

[6] The reliability of the hardware/software platform executing $\widehat{P}$

[7] Random physical phenomena (cosmic rays, etc.)

Most important factors: [1], [2], [3], [4], and [6].

## Two Serious Drawbacks

1. Unfortunately, this scheme will not work for most computerized proofs, because [1] and [2] will be overwhelming.

   Suppose the original A+H code was expressed as a program $\widehat{\mathcal{P}}$ in a PL with formal semantics:

   - The size of $\widehat{\mathcal{P}}$ would be too large for rigorous analysis.
   - The size of the proof $\widehat{\pi}$ showing the correctness of $\mathcal{P}$ would be overwhelming.

   Accordingly, both $d_1$ and $d_2$ would be seriously compromised, regardless of the remaining factors (computer reliability, etc.).

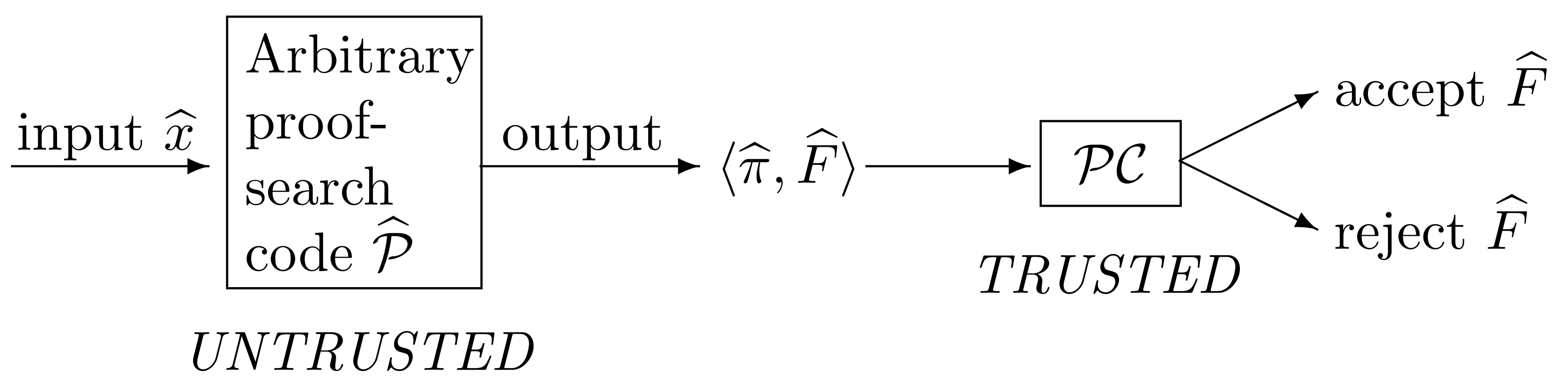
2. In addition, we would have to verify a new program $\widehat{\mathcal{P}}$ and new proof $\widehat{\pi}$ with each new project.

## A Solution

Fix a proof checker $\mathcal{PC}$ in a sufficiently rich logic. Express $\mathcal{PC}$ as a program in a language with rigorous semantics.


1. $\mathcal{PC}$ is small and simple
2. *Its* formal proof of correctness is surveyable

Now use $\mathcal{PC}$ as a filter on the output of other code:



## Advantages

Here $\widehat{\pi}_{pc}$ is a formal proof of correctness for $\mathcal{PC}$

$$
\begin{tikzcd}
\boxed{\text{START}} \arrow[d, "\mathit{eval}(A, \widehat{\pi}_{pc})"] \\
J(A, \forall\, \pi, F \;.\; [\mathcal{PC}(\pi) \hookrightarrow F] \Rightarrow\; \models F, d_1) \arrow[d, "\mathit{observe}(A, \widehat{\mathcal{PC}}(\widehat{\pi}) \hookrightarrow \widehat{F})"] \\
J(A, \mathcal{PC}(\pi) \hookrightarrow F, d_2) \arrow[d, "\mathit{infer}(\models F)"] \\
J(A, \models F, d_3)
\end{tikzcd}
$$

Now $d_1$ is high. And $d_2$ is high too, because *the size of* $\widehat{\pi}$ *is immaterial*. The size of $\widehat{F}$ is usually negligible. All we are left with are: length of computation, platform reliability, and random physical phenomena.

## Feasibility

The field of *proof engineering* has come a long way.

Several theorem-proving systems can produce certificates: HOL, Coq, Athena, etc.

4CT was recently (2005) proved in Coq.

The ultimate evidence is a low-level proof, expressible as a $\lambda$-calculus term in the type theory of Coq.

That term can be verified by the Coq proof checker, which is small and simple.

We can do even better: simplify the platform. Implement the proof-checking algorithm in silicon.

## Conclusions

4CT was *not* an epistemic landmark in mathematics.

- The concept of proof, as cognizer-independent, remains rock-solid.

Computers or not, empirical considerations are almost always involved in our justification for believing mathematical results.

Such justification is a matter of degree.

With clever engineering, computer proofs can be orders of magnitude more reliable than human-surveyed proofs.

- Clever engineering can inspire, and indeed guide, philosophy!

Basic idea: minimize our *trusted base*. We only need to trust: (1) a small and simple proof checker; and (2) the platform that executes it.

This technology is feasible. Non-trivial theorems (including 4CT) have been proved using this scheme.

**Relevant Systems For Further Reading & Study**

Arkoudas' Athena system:

http://www.cag.lcs.mit.edu/~kostas/dpls/athena

Bringsjord's (with Shilliday, Taylor, Clark, Khemlani) Slate system:

http://www.cogsci.rpi.edu/research/rair/slate