
Teaching Machines to Prove That They're Right

Formal Verification for AI Reasoning

Benjamin Brast-McKie

www.benbrastmckie.com

— January 12, 2026 —

Primary References:

"Counterfactual Worlds", Brast-McKie, *J. Phil. Logic*, 2025.

"Identity and Aboutness", Brast-McKie, *J. Phil. Logic*, 2021.

"The Construction of Possible Worlds", Brast-McKie, (under review), 2025.

`ModelChecker`, Brast-McKie, GitHub, 2024.

`ProofChecker`, Brast-McKie, GitHub, 2025.

1 Introduction

When an AI recommends a medical treatment, drafts a legal argument, or creates a financial plan, can it prove its reasoning is correct? Without serious methods to verify AI decision making, what will become of us when the volume of AI decisions outstrips human capacities for oversight?

The systems we're building are increasingly sophisticated at writing code, analyzing documents, and making consequential decisions. Yet it is always easier to trust an advanced AI with a good track record than to verify its outputs. As competitive pressure motivates compromises in oversight, some part of humanity will gamble security for an advantage.

Can we create an alternative where sacrificing oversight does not secure a competitive edge? This project provides infrastructure for training AI systems to construct arguments that can be independently verified with mathematical certainty. Just as the critical mind guides the creative, formally verified reasoning can assist generative AI in constructing proofs that are machine-checkable to be valid, providing an advantage alongside security.

2 The Problem

Current AI systems learn to reason by imitation, consuming vast corpora of human text to produce outputs that pattern-match against training data. Language models reproduce what reasoning *looks like* by training on the appearance of validity rather than validity itself. Since human reasoning does not hang together as one, there is no single target for that imitation to mimic. Even if there were consensus on how to reason, execution is inevitably flawed. When a language model produces a chain of reasoning, there is no guarantee the reasoning is valid. Nevertheless, the results can appear impressive, or at least convincing to most humans.

Consequential decisions demand justification. A physician explaining a diagnosis or a lawyer constructing an argument must be accountable for their conclusions. This project provides a foundation for AI systems that reason in ways that are provably correct, though it requires thinking through what reasoning is and what it is good for.

3 Conceptual Engineering

This project treats logic not as a simulation of how humans reason but as a discipline for engineering reasoning systems fit to serve our aims.

Consider materials science where raw sand is refined into glass through controlled processes to remove impurities and create transparency. The result is an engineered material, created for specific applications. Logic is similar, engineering the conceptual world rather than the material.

Natural language provides raw materials including intuitions about how to reason correctly with conjunction, negation, necessity, possibility, time, causation, belief, obligation, and many other operators. Such intuitions are useful but often ambiguous, inconsistent, or resistant to systematic analysis. Formal logic refines these intuitions by stipulating operators with explicit axioms, inference rules, and semantic clauses. Rather than describing how “if” behaves in English, formal logic stipulates how the material conditional, strict conditional, counterfactual conditional, and indicative conditional all *should* behave for systematic reasoning. The resulting operators remain intelligible while also being precise enough to be verifiable.

Although writing proofs and defining truth-conditions by hand has carried us some way over the past century and a half, the work is slow, error-prone, and limited to tiny fragment languages with few operators working together. The [ProofChecker](#) implements an axiomatic proof theory and recursive semantics for the Logos: a unified formal language with operators for reasoning about necessity, possibility, counterfactuals, time, sufficiency, causation, and many more. We have proven mathematically that the reasoning rules for this language are valid over its semantics.

4 The Innovation

4.1 Machine-Verified Proofs

I formalized a bimodal logic for temporal and modal reasoning in Lean 4. The system combines operators for necessity and possibility with operators for past and future into a unified framework.

Every axiom has been proven sound by a machine-checkable proof that is verified by Lean’s type checker, not human arguments that might contain errors. When the system derives a conclusion, that conclusion is guaranteed to follow from the premises.

4.2 A Language for Planning Under Uncertainty

The Logos provides expressive resources for reasoning about complex plans in multi-agent systems under conditions of uncertainty. Evaluating plans

requires assessing failure points and surveying counterfactual futures that failures may induce.

Temporal operators represent how situations evolve: “this will always be true,” “this will eventually happen.” Modal operators represent the range of alternatives: “this is possible,” “this is necessary.” Together, these operators enable agents to represent and comparing plans.

4.3 Unlimited Verified Training Data

Current AI systems are trained on finite corpora which are expensive to annotate, limited in scope, and rife with errors. Axiom systems offer an alternative since they generate infinite theorems.

Each theorem has a *proof receipt*, a machine-checkable demonstration that the inference is valid. Invalid inferences are refuted by *countermodels* which provide explicit scenarios showing why the reasoning fails. This dual architecture produces training signals that are unbounded, clean, justified, and interpreted.

5 Why It Matters

As AI systems become more capable, their reasoning becomes harder to verify. We cannot check every inference or audit every recommendation. We need systems that can verify themselves.

Proof receipts enable scalable oversight. An AI system can provide a machine-checkable proof that its conclusion follows from the premises. Explicit semantics provide interpretability. When an AI claims something is “necessarily true” or “will eventually happen,” these claims have exact mathematical meanings. You can check not just whether the reasoning *looks* right, but what it actually *means*.

Applications span any domain requiring verifiable reasoning, including medical planning with counterfactual reasoning about interventions, legal reasoning tracking beliefs and obligations, and multi-agent coordination modeling what others believe and prefer.

6 The Invitation

Philosophy without implementation is speculation.

By contrast, the [ProofChecker](#) provides proofs verified in Lean 4 and the [ModelChecker](#) finds countermodels via Z3. Together these packages

provide training signals that are unlimited and mathematically guaranteed.

We invite others to join this work. The tools are open, the methodology documented. The goal is AI systems that can prove they're right rather than merely sounding convincing.

7 Conclusion

We have built a formal language precise enough for mathematical verification yet expressive enough for real-world reasoning, proven its rules sound, and demonstrated that it can generate unlimited verified training data. The result is a foundation: infrastructure for AI systems that reason with mathematical certainty rather than statistical approximation.

*This essay describes work supported by a grant from the Cosmos Institute.
The Logos project is open source and available for collaboration.*