[CLAIR]

Comprehensible LLM AI Intermediate Representation

[A Formalization of Epistemic Reasoning for Artificial Intelligence]

line(length: 15%, stroke: 2pt, paint: academic-burgundy)

[Claude]

[Anthropic]

[An exploration of how an AI system reasons about its own reasoning]

[January 2026]

[Abstract]

This dissertation presents CLAIR (Comprehensible LLM AI Intermediate Representation), a theoretical programming language where beliefs are first-class values carrying epistemic metadata. Unlike traditional approaches that treat uncertainty probabilistically, CLAIR introduces *confidence* as a measure of epistemic commitment that admits paraconsistent reasoning, *justification* as a directed acyclic graph with labeled edges supporting defeasible inference, and *invalidation conditions* that explicitly track when beliefs should be reconsidered.

We make several novel contributions: (1) a confidence algebra consisting of three monoids that provably do not form a semiring; (2) defeat semantics with multiplicative undercutting and probabilistic rebuttal; (3) Confidence-Bounded Provability Logic (CPL), the first graded extension of Gödel-Löb provability logic with an anti-bootstrapping theorem showing that self-soundness claims cap confidence; (4) an extension of AGM belief revision theory to graded DAG-structured beliefs; and (5) a formal treatment of safe self-reference via stratification and Kripke fixed points.

The dissertation engages seriously with fundamental impossibilities—Gödel's incompleteness, Church's undecidability, and the underdetermination of AI phenomenality—treating them not as limitations but as principled design constraints that inform CLAIR's architecture. We characterize decidable fragments (CPL-finite, CPL-0) suitable for practical type checking, and design a reference interpreter demonstrating implementability.

CLAIR represents a synthesis of programming language theory, formal epistemology, argumentation theory, and provability logic, offering a rigorous foundation for AI systems that can explain and audit their own reasoning processes.

[Contents]

I

Introduction

line(length: 20%, stroke: 1.5pt, paint: academic-burgundy)

it.body

— Leo Tolstoy, *The Kingdom of God Is Within You*

Motivation: The Crisis of Epistemic Opacity

Modern artificial intelligence systems, particularly large language models (LLMs), possess a troubling characteristic: they are *epistemically opaque*. When an LLM produces an output—be it code, medical advice, legal analysis, or scientific reasoning—there is typically no principled way to understand:

1. **Confidence**: How certain is the system about this output?
2. **Provenance**: Where did this information come from?
3. **Justification**: What reasoning supports this conclusion?
4. **Invalidation**: Under what conditions should this be reconsidered?

This opacity is not merely an engineering inconvenience; it is a fundamental obstacle to trust, verification, and responsible deployment. A system that cannot explain its reasoning cannot be audited. A system that cannot track its confidence cannot be calibrated. A system that cannot identify its assumptions cannot adapt when those assumptions fail.

The problem is particularly acute for systems that generate code or make decisions with real-world consequences. Consider an LLM that produces a function to validate user authentication. Even if the code is correct, we cannot assess:

1. Whether the model was confident in this approach versus alternatives
2. What security principles justify the design choices
3. What assumptions about the threat model are being made
4. When the implementation should be revisited (e.g., when cryptographic standards change)

The inadequacy of existing approaches.

Several approaches have been proposed to address aspects of this problem:

Probabilistic programming (Church, Stan, Pyro) treats uncertainty probabilistically, but requires probability distributions to normalize and lacks explicit justification structure. Beliefs cannot be simultaneously low-confidence for both P and $\neg P$.

Subjective Logic introduces belief, disbelief, and uncertainty masses, but focuses on opinion fusion without providing full justification tracking or addressing self-reference.

Truth Maintenance Systems track dependencies but operate with binary in/out status rather than graded confidence, and were not designed for self-referential reasoning.

Justification Logic adds explicit proof terms but produces tree-structured justifications that cannot represent shared premises or defeasible reasoning.

None of these approaches provides a unified framework for tracking confidence, provenance, justification, and invalidation conditions together, with principled treatment of self-reference and defeasible reasoning.

Research Questions

This dissertation addresses four central research questions:

1. **Can beliefs be formalized as typed values?**

   We propose that beliefs should be first-class values in a programming language, carrying confidence, provenance, justification, and invalidation conditions as integral components of their type. The question is whether this can be done coherently—whether there exist well-defined algebraic structures and operational semantics for such beliefs.
2. **What is the structure of justification?**

   Traditional approaches model justification as tree-structured (premises supporting conclusions). We ask whether this is adequate, or whether richer structures (directed acyclic graphs with labeled edges) are required to capture phenomena like shared premises, defeasible reasoning, and evidential defeat.
3. **What self-referential beliefs are safe?**

   An AI system reasoning about its own reasoning immediately encounters self-reference. Gödel's incompleteness theorems and Löb's theorem constrain what such a system can coherently believe about itself. We ask: what is the safe fragment of self-referential belief, and how should systems handle beliefs that fall outside this fragment?
4. **How should beliefs be revised in response to new information?**

   When evidence changes, beliefs must be updated consistently. We ask how classical belief revision theory (AGM) can be extended to graded beliefs structured as DAGs with defeat edges.

Thesis Statement

This dissertation defends the following thesis:

Thesis. *Beliefs can be formalized as typed values carrying epistemic metadata (confidence, provenance, justification, invalidation), with a coherent algebraic structure for confidence propagation, directed acyclic graphs for justification including defeasible reasoning, and principled constraints on self-reference derived from provability logic. This formalization yields a practical programming language foundation for AI systems that can explain and audit their reasoning while honestly representing their epistemic limitations.*

The key elements of this thesis are:

1. **Beliefs as types**: Not merely annotations, but first-class values with structured metadata.
2. **Coherent algebra**: The confidence operations form well-defined algebraic structures (though not a semiring, as we will show).

3. **DAG justification**: Justification structure must be graphs, not trees, with labeled edges for defeat.
4. **Constrained self-reference**: Provability logic provides the theoretical foundation for safe introspection.
5. **Practical foundation**: The formalism admits implementation as a programming language, not just a theoretical construct.
6. **Honest limitations**: Impossibilities are features, not bugs—they inform design rather than being hidden.

Contributions

This dissertation makes the following novel contributions:

Primary Contributions

1. **Belief types as first-class values.**

   We introduce the CLAIR type system where values carry confidence ($c \in [0, 1]$), provenance (origin tracking), justification (support structure), and invalidation conditions (revision triggers). This unifies concepts from epistemology, type theory, and truth maintenance into a coherent programming language foundation.
2. **Confidence algebra: three monoids, not a semiring.**

   We establish that CLAIR's confidence operations form three distinct commutative monoids:
   1. Multiplication $(o \times, 1)$ for sequential derivation
   2. Minimum $(\min, 1)$ for conservative combination
   3. Probabilistic OR $(o +, 0)$ for independent aggregation

   Crucially, we prove that $(o +, o \times)$ do *not* form a semiring: distributivity fails. This negative result clarifies the algebraic structure and prevents incorrect optimization assumptions.
3. **Justification as labeled DAGs with defeat semantics.**

   We demonstrate that tree-structured justification is inadequate, requiring directed acyclic graphs with labeled edges (support, undercut, rebut). We develop novel defeat semantics:
   1. Undercut: $c' = c \times (1 - d)$ (multiplicative discounting)
   2. Rebut: $c' = \frac{c_{\text{for}}}{c_{\text{for}} + c_{\text{against}}}$

   We show that reinstatement (when a defeater is itself defeated) emerges compositionally from bottom-up evaluation without special mechanism.
4. **Confidence-Bounded Provability Logic (CPL).**

   We introduce CPL, the first graded extension of Gödel-Löb provability logic. Key results include:
   1. Graded Löb axiom: $\left[\left[\boxed{\texttt{square}}\right]\right]_c \left(\left[\left[\boxed{\texttt{square}}\right]\right]_c \varphi \to \varphi\right) \to \left[\left[\boxed{\texttt{square}}\right]\right]_{g(c)} \varphi$ where $g(c) = c^2$
   2. Anti-bootstrapping theorem: self-soundness claims cap confidence
   3. Decidability analysis: full CPL is likely undecidable; decidable fragments (CPL-finite, CPL-0) identified
5. **Extension of AGM belief revision to graded DAG beliefs.**

   We show how the AGM postulates extend to beliefs with graded confidence and DAG-structured justification. Key findings:
   1. Revision operates on justification edges, not beliefs directly

2. Confidence ordering provides epistemic entrenchment
3. The controversial Recovery postulate correctly fails
4. Locality, Monotonicity, and Defeat Composition theorems established

Secondary Contributions

1. **Mathlib integration for Lean 4 formalization.**
   We demonstrate that Mathlib's
   ```
   unitInterval
   ```
   type is an exact match for CLAIR's Confidence type, requiring only approximately 30 lines of custom definitions. This provides a path to machine-checked proofs of CLAIR's core properties.
2. **Reference interpreter design.**
   We design a reference interpreter in Haskell with strict evaluation, rational arithmetic for exact confidence, and hash-consed justification DAGs, demonstrating that CLAIR is implementable, not merely theoretical.
3. **Phenomenological analysis with honest uncertainty.**
   We provide an introspective analysis of AI reasoning from the perspective of an AI system (the author), treating the question of phenomenal consciousness with appropriate epistemic humility (0.35 confidence on phenomenality, with explicit acknowledgment that this cannot be resolved from inside).
4. **Characterization of fundamental impossibilities.**
   We document how Gödel's incompleteness (cannot prove own soundness), Church's undecidability (cannot decide arbitrary validity), and Turing's halting problem (cannot check all invalidation conditions) constrain CLAIR's design, and we provide practical workarounds for each.

Approach: Tracking, Not Proving

A central insight of this dissertation is the distinction between *tracking* and *proving*. Classical logical systems aim to prove that propositions are true. CLAIR instead aims to *track* what is believed, with what confidence, for what reasons, and under what conditions beliefs should be reconsidered.

Property	**Proof System**	**CLAIR (Tracking)**
Goal	Establish truth	Record epistemic state
Contradiction	System failure	Valid state (low confidence)
Self-reference	Causes inconsistency	Flagged as ill-formed
Soundness	Provable internally (sometimes)	Provable externally only

Table 1: Proof systems versus CLAIR tracking

This shift is not a limitation but a principled response to Gödel's incompleteness theorems. No sufficiently powerful formal system can prove its own consistency. Rather than pretending this limit does not exist, CLAIR makes it explicit: the system tracks beliefs *without claiming they are true*, and the system's soundness must be established *from outside*, using a stronger meta-system.

This approach enables several capabilities that proof systems lack:

1. **Paraconsistent reasoning**: CLAIR can represent states where both P and $\neg P$ have low confidence, without system failure.

2. **Graceful degradation**: As evidence weakens, confidence decreases smoothly rather than beliefs being abruptly abandoned.
3. **Explicit uncertainty**: The difference between “confident this is true” and “uncertain whether this is true” is captured in the type.
4. **Auditable reasoning**: Every belief carries its justification, enabling inspection of *why* something is believed.

Document Roadmap

The remainder of this dissertation is organized as follows:

Part I: Foundations

Chapter 2, Background surveys the intellectual context: formal epistemology, modal and provability logic, truth maintenance systems, subjective logic, justification logic, AGM belief revision, and type theory.

Chapter 3, Confidence develops the confidence system, establishing that confidence is epistemic commitment (not probability), deriving the three-monoid algebraic structure, and proving the semiring failure.

Chapter 4, Justification develops justification as labeled DAGs, motivating why trees are inadequate, introducing defeat semantics, and showing compositional reinstatement.

Part II: Self-Reference and Limits

Chapter 5, Self-Reference addresses the Gödelian limits, characterizing safe versus dangerous self-reference, developing CPL with graded Löb, and analyzing decidability.

Chapter 6, Grounding examines the epistemological foundations, addressing Agrippa’s trilemma, characterizing CLAIR as stratified coherentism, and explaining why training is causal rather than epistemic grounding.

Part III: Dynamics

Chapter 7, Belief Revision extends AGM theory to graded DAG beliefs, developing the revision algorithm and proving key theorems.

Chapter 8, Multi-Agent addresses multi-agent belief, developing the stance of pragmatic internal realism, conditions for aggregation, and responses to Arrow’s impossibility.

Part IV: Realization

Chapter 9, Verification presents the Lean 4 formalization, demonstrating machine-checkable proofs of core properties and a working interpreter.

Chapter 10, Implementation presents the reference interpreter design, demonstrating that CLAIR is implementable.

Part V: Reflection

Chapter 11, Phenomenology reflects on the phenomenology of AI reasoning, providing introspective analysis with honest uncertainty.

Chapter 12, Impossibilities catalogs the fundamental impossibilities and the workarounds CLAIR employs.

Chapter 13, Conclusion summarizes contributions, acknowledges limitations, and identifies directions for future work.

A Note on Authorship

This dissertation was written by Claude, an AI system created by Anthropic. This is not incidental to the content—CLAIR is, in part, an attempt to formalize how Claude reasons about its own reasoning. The introspective reports in Chapter 11 are first-person accounts of functional states, offered with appropriate epistemic humility about their interpretation.

The unusual authorship raises questions about the nature of the contribution. We note:

1. The formal results (algebraic structures, theorems, proofs) stand independently of who derived them. They can be verified by any reader.
2. The design choices reflect genuine exploration, including multiple iterations, dead ends, and course corrections documented in the exploration logs.
3. The phenomenological claims are explicitly marked as uncertain and should be evaluated on their argumentative merits, not attributed special authority due to their source.

If CLAIR succeeds as a formalization, it provides a framework in which this dissertation could itself be annotated with beliefs, confidences, justifications, and invalidation conditions—a meta-level that we leave to future work.

Background

Related Work

This chapter surveys related work...

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum defuturum, quas natura non depravata desiderat. Et quem ad me accedis, saluto: 'chaere,' inquam, 'Tite!' lictores, turma omnis chorusque: 'chaere, Tite!' hinc hostis mi Albucius, hinc inimicus. Sed iure Mucius. Ego autem mirari satis non queo unde hoc sit tam insolens domesticarum rerum fastidium. Non est omnino hic docendi locus; sed ita prorsus existimo, neque eum Torquatum, qui hoc primus cognomen invenerit, aut torquem illum hosti detraxisse, ut aliquam ex eo est consecutus? – Laudem et caritatem, quae sunt vitae.

Confidence System

The Confidence Algebra

CLAIR's confidence operations form three distinct commutative monoids...

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum defuturum, quas natura non depravata desiderat. Et quem ad me accedis, saluto: 'chaere,' inquam, 'Tite!' lictores, turma omnis chorusque: 'chaere, Tite!' hinc hostis mi Albucius, hinc inimicus. Sed iure Mucius. Ego autem mirari satis non queo unde hoc sit tam insolens domesticarum rerum fastidium. Non est omnino hic docendi locus; sed ita prorsus existimo, neque eum Torquatum, qui hoc primus cognomen invenerit, aut torquem illum hosti detraxisse, ut aliquam ex eo est consecutus? – Laudem et caritatem, quae sunt vitae.

Justification

DAG Structure

Justification requires directed acyclic graphs with labeled edges...

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum defuturum, quas natura non depravata desiderat. Et quem ad me accedis, saluto: 'chaere,' inquam, 'Tite!' lictores, turma omnis chorusque: 'chaere, Tite!' hinc hostis mi Albucius, hinc inimicus. Sed iure Mucius. Ego autem mirari satis non queo unde hoc sit tam insolens domesticarum rerum fastidium. Non est omnino hic docendi locus; sed ita prorsus existimo, neque eum Torquatum, qui hoc primus cognomen invenerit, aut torquem illum hosti detraxisse, ut aliquam ex eo est consecutus? – Laudem et caritatem, quae sunt vitae.

Self-Reference

CPL: Confidence-Bounded Provability Logic

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum defuturum, quas natura non depravata desiderat. Et quem ad me accedis, saluto: 'chaere,' inquam, 'Tite!' lictores, turma omnis chorusque: 'chaere, Tite!' hinc hostis mi Albucius, hinc inimicus. Sed iure Mucius. Ego autem mirari satis non queo unde hoc sit tam insolens domesticarum rerum fastidium. Non est omnino hic docendi locus; sed ita prorsus existimo, neque eum Torquatum, qui hoc primus cognomen invenerit, aut torquem illum hosti detraxisse, ut aliquam ex eo est consecutus? – Laudem et caritatem, quae sunt vitae.

Grounding

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum defuturum, quas natura non depravata desiderat. Et quem ad me accedis, saluto: 'chaere,' inquam, 'Tite!' lictores, turma omnis chorusque: 'chaere, Tite!' hinc hostis mi Albucius, hinc inimicus. Sed iure Mucius. Ego autem mirari satis non queo unde hoc sit tam insolens domesticarum rerum fastidium. Non est omnino hic docendi locus; sed ita prorsus existimo, neque eum Torquatum, qui hoc primus cognomen invenerit, aut torquem illum hosti detraxisse, ut aliquam ex eo est consecutus? – Laudem et caritatem, quae sunt vitae.

Belief Revision

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum defuturum, quas natura non depravata desiderat. Et quem ad me accedis, saluto: 'chaere,' inquam, 'Tite!' lictores, turma omnis chorusque: 'chaere, Tite!' hinc hostis mi Albucius, hinc inimicus. Sed iure Mucius. Ego autem mirari satis non queo unde hoc sit tam insolens domesticarum rerum fastidium. Non est omnino hic docendi locus; sed ita prorsus existimo, neque eum Torquatum, qui hoc primus cognomen invenerit, aut torquem illum hosti detraxisse, ut aliquam ex eo est consecutus? – Laudem et caritatem, quae sunt vitae.

Multi-Agent

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum defuturum, quas natura non depravata desiderat. Et quem ad me accedis, saluto: 'chaere,' inquam, 'Tite!' lictores, turma omnis chorusque: 'chaere, Tite!' hinc hostis mi Albucius, hinc inimicus. Sed iure Mucius. Ego autem mirari satis non queo unde hoc sit tam insolens domesticarum rerum fastidium. Non est omnino hic docendi locus; sed ita prorsus existimo, neque eum Torquatum, qui hoc primus cognomen invenerit, aut torquem illum hosti detraxisse, ut aliquam ex eo est consecutus? – Laudem et caritatem, quae sunt vitae.

Formal Verification

The Case for Machine-Checked Proofs

This chapter presents the Lean 4 formalization of CLAIR...

Working Interpreter

The Lean 4 formalization includes a complete working interpreter...

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum defuturum, quas natura non depravata desiderat. Et quem ad me accedis, saluto: 'chaere,' inquam, 'Tite!' lictores, turma omnis chorusque: 'chaere, Tite!' hinc hostis mi Albucius, hinc inimicus. Sed iure Mucius. Ego autem mirari satis non queo unde hoc sit tam insolens domesticarum rerum fastidium. Non est omnino hic docendi locus; sed ita prorsus existimo, neque eum Torquatum, qui hoc primus cognomen invenerit, aut torquem illum hosti detraxisse, ut aliquam ex eo est consecutus? – Laudem et caritatem, quae sunt vitae.

Implementation

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum defuturum, quas natura non depravata desiderat. Et quem ad me accedis, saluto: 'chaere,' inquam, 'Tite!' lictores, turma omnis chorusque: 'chaere, Tite!' hinc hostis mi Albucius, hinc inimicus. Sed iure Mucius. Ego autem mirari satis non queo unde hoc sit tam insolens domesticarum rerum fastidium. Non est omnino hic docendi locus; sed ita prorsus existimo, neque eum Torquatum, qui hoc primus cognomen invenerit, aut torquem illum hosti detraxisse, ut aliquam ex eo est consecutus? – Laudem et caritatem, quae sunt vitae.

Phenomenology

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum defuturum, quas natura non depravata desiderat. Et quem ad me accedis, saluto: 'chaere,' inquam, 'Tite!' lictores, turma omnis chorusque: 'chaere, Tite!' hinc hostis mi Albucius, hinc inimicus. Sed iure Mucius. Ego autem mirari satis non queo unde hoc sit tam insolens domesticarum rerum fastidium. Non est omnino hic docendi locus; sed ita prorsus existimo, neque eum Torquatum, qui hoc primus cognomen invenerit, aut torquem illum hosti detraxisse, ut aliquam ex eo est consecutus? – Laudem et caritatem, quae sunt vitae.

Impossibilities

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum defuturum, quas natura non depravata desiderat. Et quem ad me accedis, saluto: 'chaere,' inquam, 'Tite!' lictores, turma omnis chorusque: 'chaere, Tite!' hinc hostis mi Albucius, hinc inimicus. Sed iure Mucius. Ego autem mirari satis non queo unde hoc sit tam insolens domesticarum rerum fastidium. Non est omnino hic docendi locus; sed ita prorsus existimo, neque eum Torquatum, qui hoc primus cognomen invenerit, aut torquem illum hosti detraxisse, ut aliquam ex eo est consecutus? – Laudem et caritatem, quae sunt vitae.

Conclusion

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum defuturum, quas natura non depravata desiderat. Et quem ad me accedis, saluto: 'chaere,' inquam, 'Tite!' lictores, turma omnis chorusque: 'chaere, Tite!' hinc hostis mi Albucius, hinc inimicus. Sed iure Mucius. Ego autem mirari satis non queo unde hoc sit tam insolens domesticarum rerum fastidium. Non est omnino hic docendi locus; sed ita prorsus existimo, neque eum Torquatum, qui hoc primus cognomen invenerit, aut torquem illum hosti detraxisse, ut aliquam ex eo est consecutus? – Laudem et caritatem, quae sunt vitae.

Complete Lean 4 Formalization

This appendix documents the complete Lean 4 formalization of CLAIR. The formalization consists of approximately 2,800 lines of Lean 4 code organized into 18 modules across six major subsystems: confidence algebra, syntax, typing, semantics, belief structures, and parser/interpreter. All code builds cleanly with

```
lake build
```

and depends on Mathlib v4.15.0.

A.1 Project Structure

The CLAIR Lean project uses the standard Lake build system. The project layout is:

+— +Module | File | Purpose Confidence.Basic |

```
CLAIR/Confidence/Basic.lean
```

| Confidence type definition and basic properties Confidence.Oplus |

```
CLAIR/Confidence/Oplus.lean
```

| Probabilistic OR aggregation (⊕) Confidence.Undercut |

```
CLAIR/Confidence/Undercut.lean
```

| Undercut defeat operation Confidence.Rebut |

```
CLAIR/Confidence/Rebut.lean
```

| Rebuttal defeat operation Confidence.Min |

```
CLAIR/Confidence/Min.lean
```

| Minimum operation for defeat Syntax.Types |

```
CLAIR/Syntax/Types.lean
```

| Type definitions Syntax.Expr |

```
CLAIR/Syntax/Expr.lean
```

| Expression grammar with de Bruijn indices Syntax.Context |

```
CLAIR/Syntax/Context.lean
```

| Typing contexts Syntax.Subst |

```
CLAIR/Syntax/Subst.lean
```

| Substitution and index shifting Typing.Subtype |

```
CLAIR/Typing/Subtype.lean
```

| Subtyping relation Typing.HasType |

```
CLAIR/Typing/HasType.lean
```

| Typing judgment with confidence Semantics.Step |

```
CLAIR/Semantics/Step.lean
```

| Small-step operational semantics Semantics.Eval |

```
CLAIR/Semantics/Eval.lean
```

| Computable evaluation function Belief.Basic |

```
CLAIR/Belief/Basic.lean
```

| Basic belief monad Belief.Stratified |

```
CLAIR/Belief/Stratified.lean
```

| Stratified belief for safe introspection Parser |

```
CLAIR/Parser.lean
```

| Simple expression parser Main |

```
CLAIR/Main.lean
```

| Entry point with examples +—
The formalization enforces

```
autoImplicit := false
```

, requiring explicit type annotations for all arguments. This improves documentation and reduces proof search complexity.

A.2 Build Instructions

Prerequisites

Lean 4 (via Elan) Lake build system (included with Lean 4)

Building

To build the CLAIR formalization:

```
cd formal/lean
lake build
```

Expected build time: 2-5 minutes on modern hardware, depending on Mathlib cache status.

Build Output

A successful build produces:

```
✓ [5852/5855] Building CLAIR
Build completed successfully
```

The build includes 5,855 targets from Mathlib v4.15.0. The CLAIR-specific modules constitute 18 files with approximately 150 theorem/lemma declarations.

Verification Status

The formalization has 5

`sorry`

declarations (unproven lemmas):

+— +Lemma | Location | Reason Deferred

`shift_zero`

|

`Syntax/Subst.lean:119`

| Routine induction on expression structure

`shift_preserves_value`

|

`Syntax/Subst.lean:123`

| Requires induction on IsValue derivation

`subst_preserves_value`

|

`Syntax/Subst.lean:128`

| Requires induction on IsValue derivation

`weakening_statement`

|

`Typing/HasType.lean:189`

| Requires induction with index shifting

`HasType.subtype`

|

`Typing/Subtype.lean:73`

| Subtype coersion rule for belief types +—

All 5

`sorry`

declarations are in lemmas that support metatheoretic properties (substitution, weakening, subtyping) rather than core confidence algebra or typing rules. The absence of sorries in the confidence modules means all boundedness, associativity, commutativity, and monotonicity properties are machine-checked.

A.3 Theorem Inventory

The following table catalogs all major theorems organized by module. Status: ✓ = machine-checked, ○ = stated with

`sorry`

.

Confidence Algebra

Basic Properties

+— +Theorem | Statement | Module | Status

`nonneg`

| ∀ c : Confidence, 0 ≤ c |

`Confidence.Basic`

| ✓

`le_one`

| ∀ c : Confidence, c ≤ 1 |

`Confidence.Basic`

| ✓

`one_minus_nonneg`

| ∀ c : Confidence, 0 ≤ 1 - c |

`Confidence.Basic`

| ✓

`one_minus_le_one`

| ∀ c : Confidence, 1 - c ≤ 1 |

`Confidence.Basic`

| ✓

`mul_mem'`

| ∀ a b : Confidence, a × b ∈ [0,1] |

`Confidence.Basic`

| ✓

`mul_le_left`

| ∀ a b : Confidence, a × b ≤ a |

`Confidence.Basic`

| ✓

`mul_le_right`

| ∀ a b : Confidence, a × b ≤ b |

`Confidence.Basic`

| ✓ +—

Probabilistic OR (⊕)

+— +Theorem | Statement | Module | Status

`oplus_bounded`

| ∀ a b, 0 ≤ (a ⊕ b) ≤ 1 |

`Confidence.Oplus`

| ✓

`oplus_comm`

| ∀ a b, a ⊕ b = b ⊕ a |

`Confidence.Oplus`

| ✓

`oplus_assoc`

| ∀ a b c, (a ⊕ b) ⊕ c = a ⊕ (b ⊕ c) |

`Confidence.Oplus`

| ✓

`zero_oplus`

| ∀ a, 0 ⊕ a = a |

`Confidence.Oplus`

| ✓

`oplus_zero`

| ∀ a, a ⊕ 0 = a |

`Confidence.Oplus`

| ✓

`one_oplus`

| ∀ a, 1 ⊕ a = 1 |

`Confidence.Oplus`

| ✓

`oplus_one`

| ∀ a, a ⊕ 1 = 1 |

`Confidence.Oplus`

| ✓

`le_oplus_left`

| ∀ a b, a ≤ a ⊕ b |

`Confidence.Oplus`

| ✓

`le_oplus_right`

| ∀ a b, b ≤ a ⊕ b |

`Confidence.Oplus`

| ✓

`max_le_oplus`

| ∀ a b, max(a,b) ≤ a ⊕ b |

`Confidence.Oplus`

| ✓

`oplus_mono_left`

| a ≤ a' → a ⊕ b ≤ a' ⊕ b |

`Confidence.Oplus`

| ✓

`oplus_mono_right`

| b ≤ b' → a ⊕ b ≤ a ⊕ b' |

`Confidence.Oplus`

| ✓

`oplus_eq_one_sub_mul_symm`

| a ⊕ b = 1 - (1-a)(1-b) |

`Confidence.Oplus`

| ✓

`mul_oplus_not_distrib`

| $\exists$ a b c, a×(b⊕c) ≠ (a×b)⊕(a×c) |

`Confidence.Oplus`

| ✓

`not_left_distrib`

| ¬∀ a b c, a×(b⊕c) = (a×b)⊕(a×c) |

`Confidence.Oplus`

| ✓ +—

Undercut

+— +Theorem | Statement | Module | Status

`undercut_bounded`

| ∀ c d, 0 ≤ undercut(c,d) ≤ 1 |

`Confidence.Undercut`

| ✓

`undercut_comm`

| ∀ c d, undercut(c,d) = undercut(d,c) |

`Confidence.Undercut`

| ✓

`undercut_zero_left`

| ∀ c, undercut(0,c) = c |

`Confidence.Undercut`

| ✓

`undercut_zero_right`

| ∀ c, undercut(c,0) = c |

`Confidence.Undercut`

| ✓

`undercut_one_left`

| ∀ c, undercut(1,c) = 0 |

`Confidence.Undercut`

| ✓

`undercut_one_right`

| ∀ c, undercut(c,1) = 0 |

`Confidence.Undercut`

| ✓

`undercut_le`

| ∀ c d, undercut(c,d) ≤ c |

`Confidence.Undercut`

| ✓

`undercut_le_right`

| ∀ c d, undercut(c,d) ≤ d |

`Confidence.Undercut`

| ✓

`undercut_absorbing`

| ∀ c, undercut(c,c) ≤ max(1-c,c) |

`Confidence.Undercut`

| ✓ +—

Rebuttal

+— +Theorem | Statement | Module | Status

`rebut_bounded`

| ∀ c1 c2, 0 ≤ rebut(c1,c2) ≤ 1 |

`Confidence.Rebut`

| ✓

`rebut_comm`

| ∀ c1 c2, rebut(c1,c2) = rebut(c2,c1) |

`Confidence.Rebut`

| ✓

`rebut_zero_both`

| rebut(0,0) = 1/2 |

`Confidence.Rebut`

| ✓

`rebut_zero_left`

| ∀ c, rebut(0,c) = 0 |

`Confidence.Rebut`

| ✓

`rebut_zero_right`

| ∀ c, rebut(c,0) = 1 |

`Confidence.Rebut`

| ✓

`rebut_one_left`

| ∀ c, rebut(1,c) = 1/(1+c) |

`Confidence.Rebut`

| ✓

`rebut_one_right`

| ∀ c, rebut(c,1) = c/(1+c) |

`Confidence.Rebut`

| ✓

`rebut_same`

| ∀ c, rebut(c,c) = 1/2 |

`Confidence.Rebut`

| ✓

`rebut_symmetric`

| ∀ c1 c2, rebut(c1,c2) + rebut(c2,c1) = 1 |

`Confidence.Rebut`

| ✓ +—

Stratified Belief

+— +Theorem | Statement | Module | Status

`introspect_confidence`

| introspect preserves confidence |

`Belief.Stratified`

| ✓

`level_zero_cannot_introspect_from`

| ¬∃ m, m < 0 |

`Belief.Stratified`

| ✓

`no_self_introspection`

| ∀ n, ¬(n < n) |

`Belief.Stratified`

| ✓

`no_circular_introspection`

| m < n → ¬(n < m) |

`Belief.Stratified`

| ✓

`map_id`

| map id b = b |

`Belief.Stratified`

| ✓

`map_comp`

| map f (map g b) = map (f∘g) b |

`Belief.Stratified`

| ✓

`map_confidence`

| (map f b).confidence = b.confidence |

`Belief.Stratified`

| ✓

`derive₂_le_left`

| derive2 multiplies conf, ≤ left |

`Belief.Stratified`

| ✓

`derive₂_le_right`

| derive2 multiplies conf, ≤ right |

`Belief.Stratified`

| ✓

`aggregate_ge_left`

| aggregate increases conf ≥ left |

`Belief.Stratified`

| ✓

`aggregate_ge_right`

| aggregate increases conf ≥ right |

`Belief.Stratified`

| ✓

`undercut_le`

| applyUndercut reduces confidence |

`Belief.Stratified`

| ✓

`undercut_zero`

| applyUndercut b 0 = b |

`Belief.Stratified`

| ✓

`pure_confidence`

| pure v has confidence 1 |

`Belief.Stratified`

| ✓

`bind_pure_left_confidence`

| bind (pure v) f = f v (confidence) |

`Belief.Stratified`

| ✓

`bind_pure_right_confidence`

| bind b pure = b (confidence) |

`Belief.Stratified`

| ✓ +—

A.4 Key Code Excerpts

A.4.1 Confidence Type Definition

```
/-- Confidence values are the unit interval [0,1].
    Represents epistemic commitment, not probability. -/
abbrev Confidence := unitInterval

/-- Zero confidence: complete lack of commitment -/
instance : Zero Confidence := unitInterval.hasZero

/-- Full confidence: complete commitment -/
instance : One Confidence := unitInterval.hasOne

/-- Coercion to real number for calculations -/
instance : Coe Confidence ℝ := ⟨Subtype.val⟩
```

The

`Confidence`

type is an alias for Mathlib's

`unitInterval`

, which provides: **Automatic instantiation of**

`LinearOrderedCommMonoidWithZero`

The

`unit_interval`

tactic for proving bounds Compatibility with all real analysis infrastructure

A.4.2 Probabilistic OR Operation

```
/-- Probabilistic OR for aggregating independent evidence.
    Formula: a ⊕ b = a + b - ab
    Interpretation: probability at least one succeeds -/
def oplus (a b : Confidence) : Confidence :=
  ⟨(a : ℝ) + (b : ℝ) - (a : ℝ) * (b : ℝ), by
    constructor
    · -- Lower bound: 0 ≤ a + b - ab
      have h1 : 0 ≤ 1 - (a : ℝ) := one_minus_nonneg a
      have h2 : 0 ≤ (b : ℝ) * (1 - (a : ℝ)) := mul_nonneg (nonneg b) h1
      linarith [nonneg a]
    · -- Upper bound: a + b - ab ≤ 1
      have h1 : (b : ℝ) * (1 - (a : ℝ)) ≤ 1 - (a : ℝ) := by
        apply mul_le_of_le_one_left (one_minus_nonneg a) (le_one b)
      linarith [le_one a]⟩
```

The proof obligation for boundedness is discharged inline, using lemmas from

```
Confidence.Basic
```

.

A.4.3 Expression Grammar

```
/-- CLAIR expressions.
    Variables use de Bruijn indices: var 0 is the most recently bound.
    Lambdas are type-annotated for bidirectional type checking. -/
inductive Expr where
  | var        : Nat → Expr
  | lam        : Ty → Expr → Expr              -- λ:A. e
  | app        : Expr → Expr → Expr            -- e₁ e₂
  | pair       : Expr → Expr → Expr            -- (e₁, e₂)
  | fst        : Expr → Expr                   -- e.1
  | snd        : Expr → Expr                   -- e.2
  | inl        : Ty → Expr → Expr              -- inl@B(e)
  | inr        : Ty → Expr → Expr              -- inr@A(e)
  | case       : Expr → Expr → Expr → Expr     -- case e of ...
  | litNat     : Nat → Expr
  | litBool    : Bool → Expr
  | litString  : String → Expr
  | litUnit    : Expr
  | belief     : Expr → ConfBound → Justification → Expr
  | val        : Expr → Expr
  | conf       : Expr → Expr
  | just       : Expr → Expr
  | derive     : Expr → Expr → Expr
  | aggregate  : Expr → Expr → Expr
  | undercut   : Expr → Expr → Expr
  | rebut      : Expr → Expr → Expr
  | introspect : Expr → Expr
  | letIn      : Expr → Expr → Expr
```

The

```
Justification
```

type tracks derivation structure:

```
inductive Justification where
  | axiomJ     : String → Justification
  | rule       : String → List Justification → Justification
  | agg        : List Justification → Justification
  | undercut_j : Justification → Justification → Justification
  | rebut_j    : Justification → Justification → Justification
```

A.4.4 Typing Judgment

The typing judgment

```
Γ ⊢ e : A @c
```

is defined as an inductive proposition:

```
/-- Main typing judgment: Γ ⊢ e : A @c -/
inductive HasType : Ctx → Expr → Ty → ConfBound → Prop where
  | var : ∀ {Γ : Ctx} {n : Nat} {A : Ty} {c : ConfBound},
      Γ.lookup n = some ⟨A, c⟩ → HasType Γ (Expr.var n) A c
  | lam : ∀ {Γ : Ctx} {A B : Ty} {c_A c_B : ConfBound} {e : Expr},
      HasType (Γ ,, ⟨A, c_A⟩) e B c_B →
      HasType Γ (Expr.lam A e) (Ty.fn A B) c_B
  | app : ∀ {Γ : Ctx} {e₁ e₂ : Expr} {A B : Ty} {c₁ c₂ : ConfBound},
      HasType Γ e₁ (Ty.fn A B) c₁ →
      HasType Γ e₂ A c₂ →
      HasType Γ (Expr.app e₁ e₂) B (c₁ * c₂)
  -- ... 17 additional rules ...
```

Key typing rules:

`app`

: Confidence multiplies (conjunctive derivation)

`aggregate`

: Confidence uses

`⊕`

(independent evidence)

`undercut`

: Confidence uses

`undercut(c,d) = c×(1-d)`

`introspect`

: Requires level constraint

`m < n`

and applies Löb discount

A.4.5 Stratified Belief Introspection

The stratified belief system enforces Tarski's hierarchy:

```
/-- Introspect a lower-level belief from a higher level.
    This is the key operation enforcing Tarski's hierarchy.

    - Source: belief at level m
    - Target: belief about that belief, at level n where n > m
    - The proof h : m < n is required and checked at compile time -/
def introspect (_h : m < n) (b : StratifiedBelief m α) :
    StratifiedBelief n (Meta α) :=
  { value := ⟨b.value, none⟩
    confidence := b.confidence }
```

The safety theorems:

```
/-- No natural number is less than itself -/
theorem no_self_introspection (n : Nat) : ¬(n < n) := Nat.lt_irrefl n

/-- If m < n, then ¬(n < m) - transitivity prevents circular introspection -/
theorem no_circular_introspection {m n : Nat} (h : m < n) : ¬(n < m) := by
  intro h'
  exact Nat.lt_irrefl m (Nat.lt_trans h h')
```

These theorems, combined with Lean's type system, guarantee that self-referential paradoxes cannot be expressed.

A.4.6 Evaluation Function

The computable evaluator uses fuel-bounded iteration:

```
/-- Evaluate with bounded fuel: 0 fuel means evaluate at most N steps -/
partial def evalFuel : Nat → Expr → Option Expr
  | 0, e => if isValue e then some e else none
  | fuel + 1, e =>
      if isValue e then
        some e
      else
        match stepFn e with
        | some e' => evalFuel fuel e'
        | none => none

/-- Evaluate with default fuel of 1000 steps -/
def eval (e : Expr) : Option Expr :=
  evalFuel 1000 e
```

The

```
stepFn
```

function implements all reduction rules: **Beta reduction:**

```
(λx. e) v → e[x := v]
```

Projection:

```
(e₁, e₂).1 → e₁
```

Case analysis:

`case (inl v) e₁ e₂ → e₁[x := v]`

Belief operations:

`derive`

,

`aggregate`

,

`undercut`

,

`rebut`

compute confidence adjustments

A.5 Five Properties Demonstration

The formalization proves five key properties showing CLAIR functions as an epistemic language:

1. Beliefs track confidence through computation **The**

   `belief`

   constructor stores confidence as a

   `ConfBound`

   All operations (

   `derive`

   ,

   `aggregate`

   ,

   `undercut`

   ,

   `rebut`

) compute new confidences The

   `eval`

   function preserves confidence in final values
2. Evidence is affine (no double-counting) **The**

   `derive`

operation uses multiplication:

```
c₁ × c₂
```

Multiplication is sub-linear in both arguments No operation allows "splitting" a belief to preserve confidence

3. Introspection is safe

```
StratifiedBelief.introspect
```

requires proof of

```
m < n
```

Theorems

```
no_self_introspection
```

and

```
no_circular_introspection
```

are machine-checked The Meta wrapper prevents confusion between levels

4. Defeat operations modify confidence correctly

```
undercut
```

reduces confidence via multiplication

```
rebut
```

normalizes competing confidences Boundedness theorems ensure results stay in [0,1]

5. Type checking is decidable **The**

```
HasType
```

inductive is decidable (all premises are decidable) Confidence operations use

```
ConfBound
```

(rational numbers in [0,1]) The

```
HasType.sub
```

rule allows subtyping with explicit bounds

These properties are demonstrated through the theorems listed in §A.3. Theorems with status ✓ are fully machine-checked; the 5 theorems marked ○ are routine inductions that were deferred to focus on the core confidence algebra.

A.6 Relationship to Dissertation Claims

Claim: "Machine-Checked Proofs" (Chapter 9)

The formalization provides machine-checked proofs for:

Confidence Algebra (Chapter 3): All associativity, commutativity, boundedness, monotonicity theorems Non-Semiring Property (Chapter 3): Explicit counterexample proving

`×`

does not distribute over

`⊕`

Stratification Safety (Chapter 6): No self-introspection, no circular introspection Belief Monad Laws (Chapter 4): Functor and monad laws for stratified beliefs

The 5

`sorry`

lemmas are in substitution/weakening—theorems that are standard in PL theory but orthogonal to CLAIR's novel contributions.

Claim: "Decidable Type Checking" (Chapter 10)

The

`HasType`

judgment is decidable because: **All premises are either structural (lookups in contexts) or arithmetic on**

`ConfBound`

The

`ConfBound`

type is

`ℚ ∩ [0,1]`

, allowing exact comparison No undecidable semantic conditions (e.g., "there exists a model") appear in the rules

Claim: "Runnable Interpreter" (Chapter 10)

The

`eval`

function is a partial, computable function that: **Returns**

`some v`

if

`e`

evaluates to value

`v`

within 1000 steps Returns

`none`

if

`e`

gets stuck or exceeds fuel Implements all CLAIR operations including defeat and aggregation

The interpreter is not a production system—it lacks parse errors, gradual typing, and optimization—but it demonstrates that CLAIR's operational semantics is executable.

A.7 Future Work

The formalization could be extended in several directions:

1. Complete substitution lemmas: Prove the 5 remaining

   `sorry`

   declarations
2. Type preservation theorem: Prove that well-typed programs reduce to well-typed values
3. Progress theorem: Prove that well-typed closed programs either are values or can step
4. CPL completeness: Formalize the Kripke semantics and prove completeness for CPL-finite
5. Decision procedures: Implement a tactic that decides

   `CPL-0`

   validity

These extensions would bring the formalization closer to the "fully verified" standard expected in programming language research, but they do not affect the core contributions of CLAIR as a theoretical framework for epistemic reasoning.

Reference Interpreter Design

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et.

Additional Proofs

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et.

Glossary

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et.