

Hybrid Gödel-Fitch Approach via Scope-Constrained Reinforcement Learning: Integrating Certified Verification with Inductive Reasoning

Research Proposal Specification

February 25, 2026

Abstract

Automated Theorem Proving (ATP) in recursive domains like Peano Arithmetic (PA) faces dual challenges: search space explosion and logical grounding issues. This research proposes a neuro-symbolic framework integrating Fitch-style Natural Deduction with Gödel-inspired Prime-Factor Embedding (PFE). Crucially, we incorporate the Rocq (Coq) Proof Assistant as a certified verification oracle and data generator. We introduce a Bi-directional Translation Layer (Parser) to bridge visual Fitch proofs and Rocq's formal type theory. By implementing a curriculum-based training pipeline guided by certified data, this system aims to achieve high search efficiency and generate human-readable, formally verified proofs.

1 Introduction

The automation of mathematical reasoning, particularly in structural induction, remains a significant challenge. Purely neural approaches often lack a formal verification layer, leading to logically unsound results despite plausible appearances. Conversely, traditional symbolic ATPs guarantee correctness but struggle with scalability and human interpretability.

This research bridges the gap by grounding the ATP process in Fitch-style Natural Deduction, providing a visual and hierarchical structure for reasoning. We explicitly model Fitch scopes to prune the search space for Reinforcement Learning (RL) agents. Furthermore, we integrate the Rocq Proof Assistant to serve as the ultimate gold standard for verification, ensuring that every generated proof is not only intuitively correct but also mechanically certified against the rigorous standards of the Calculus of Inductive Constructions (CiC).

2 System Architecture and Theoretical Framework

2.1 Overall Research Pipeline

Our system operates in three integrated phases: Offline Data Generation via Rocq, Online RL Training via the Fitch Environment, and Certified Verification via the Rocq Kernel.

2.2 Scope-Constrained Logic: Fitch-style System

Fitch-style deduction utilizes nested boxes to denote subproofs, defining strict accessibility rules for premises. Our Fitch Symbolic Environment enforces these rules during exploration, acting as a fast filter for invalid logical moves. A premise is accessible only if it resides in the current scope or a direct ancestor scope, preventing assumptions from leaking into global contexts.

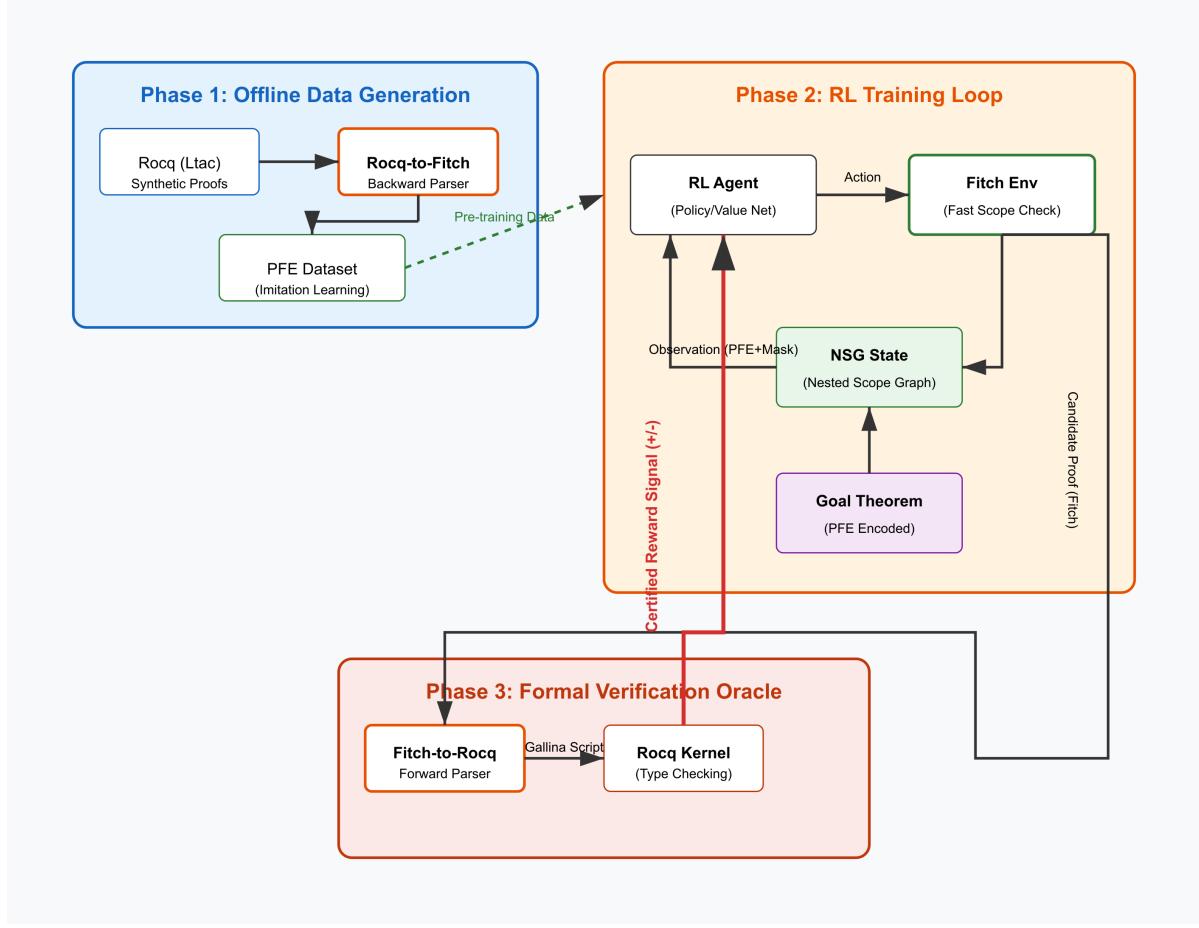


Figure 1: Integrated Research Architecture Flowchart. Phase 1 generates synthetic data using Rocq. Phase 2 is the main RL loop where the agent interacts with the Fitch environment. Phase 3 provides certified feedback by translating Fitch proofs back into Rocq for type-checking.

2.3 Arithmetization: Prime-Factor Embedding (PFE)

To enable neural networks to perceive logical structure, we map symbols to unique primes (e.g., universal quantification to 7, addition to 31). Formulas are encoded into high-dimensional vectors representing the structural depth of their Gödel numbers. This provides a unique arithmetic signature for every logical state.

3 Formal Translation Layer: The Bi-directional Parser

A critical innovation is the Bi-directional Parser that bridges the semantic gap between visual Fitch structures and Rocq’s formal Gallina language.

3.1 Backward Parser: Rocq to Fitch

Used in Phase 1 for data generation. It deconstructs Rocq tactic scripts (e.g., intros, induction) and maps them to equivalent Fitch scope operations. For instance, an induction tactic is decomposed into the opening of a base case subproof and an inductive step subproof, generating gold standard PFE sequences for imitation learning.

3.2 Forward Parser: Fitch to Rocq

Used in Phase 3 for certification. It translates agent-generated Fitch proofs into runnable Gallina scripts. A Fitch subproof assuming a property $P(k)$ is translated into a Rocq section requiring intros k Hk . Success in Rocq's type-checking provides the final certified reward signal.

4 Methodology: Curriculum and Execution

4.1 Curriculum Learning Strategy

The agent is trained through a four-stage curriculum, transitioning from basic grammar to autonomous discovery:

1. Stage 1: Propositional Grammar. Mastering subproof boxes for basic connectives.
2. Stage 2: Predicate Logic. Handling quantifiers and variable substitution via PFE.
3. Stage 3: Structural Induction. Instantiating the induction template (Base Case / Inductive Step).
4. Stage 4: Peano Synthesis. Integrating PA axioms for complex arithmetic proofs.

4.2 Certified Execution Loop

Algorithm 1 Hybrid Gödel-Fitch-Rocq System Loop

```
1: Input: Target Theorem T, Rocq Kernel K
2: env ← Initialize Fitch_Environment(T)
3: while not converged do
4:   mask ← env.Get_Accessibility_Mask()                                ▷ Fitch Scoping
5:   state ← Build_NSG(env.history)
6:   action ← Sample(Policy(state, mask))
7:   done ← env.Step(action)
8:   if done and env.proven then
9:     rocq_script ← Forward_Parser(env.history)
10:    verified ← K.Check(rocq_script)
11:    if verified then Reward ← 100                                     ▷ Certified Success
12:    else Reward ← -50                                         ▷ Logical Gap
13:   end if
14: end if
15: end while
```

5 Conclusion

By fusing Gödel's arithmetization, Fitch's structural scoping, and Rocq's certified verification, this research proposes a robust framework for neuro-symbolic reasoning. This approach ensures that the agent masters the grammar of logic before attempting complex arithmetic discovery.

References

- [1] F. B. Fitch, Symbolic Logic, 1952.
- [2] The Rocq Development Team, The Rocq Proof Assistant, 2024.