Case Study: Proving $\sqrt{2}$ Irrational with LPTP and an LLM

Fred Mesnard   Étienne Payet   Wim Vanhoof

LIM, Université de La Réunion, France
Université de Namur, Belgium

ICLP 2026

Motivation

A pure Prolog program over Peano numerals (0, s/1):

```
nat(0).
nat(s(X)) :- nat(X).

plus(0,Y,Y).
plus(s(X),Y,s(Z)) :- plus(X,Y,Z).
```

```
times(0,Y,0).
times(s(X),Y,Z) :-
    times(X,Y,P), plus(P,Y,Z).

gcd(X,Y,D) :-
  ( leq(X,Y) -> gcd_leq(X,Y,D)
  ; gcd_leq(Y,X,D)).
```

Query: Find two coprime naturals p, q such that $p^2 = 2q^2$.

```
?- nat(S), plus(P,Q,S), gcd(P,Q,s(0)),
times(P,P,P2), times(Q,Q,Q2), plus(Q2,Q2,P2).
```

- no resolution-based engine, Prolog included, can return "no"
- **Goal:** Formally prove $\sqrt{2} \notin \mathbb{Q}$ inside an LP framework, with LLM help

LPTP in a Nutshell (1/2)

- **LPTP** (Logic Program Theorem Prover) by Stärk [1998]: prove properties of normal Prolog programs.
- For each user-defined predicate R, three predicates: R^s (success), R^f (failure), R^t (termination).
- Operators $\mathcal{S}$, $\mathcal{F}$, $\mathcal{T}$ map Prolog goals to first-order formulas.
- The inductive extension $\mathrm{IND}(P)$ provides:
  1. Clark's equality axioms
  2. Groundness axioms for $gr/1$
  3. Uniqueness & totality: $\neg(R^s \wedge R^f)$, $\quad R^t \rightarrow R^s \vee R^f$
  4. Fixed-point axioms (Clark's completion)
  5. Induction along predicate definitions

LPTP in a Nutshell (2/2)

- Proofs are written in **natural deduction** $\Rightarrow$ human readable
- Specification language $\hat{\mathcal{L}}$ extends first-order logic; can add new function/predicate symbols
- Interactive proof checker: validates each derivation step; reports the *first* incorrect step
- Tactic `by gap`: admit a step $\Rightarrow$ allows incremental, top-down development

Key property [Stärk]: $\mathrm{IND}(P)$ is always consistent

Once proof-checked, a lemma can safely be added to the library

The Classic Proof of $\sqrt{2} \notin \mathbb{Q}$

Proof by contradiction

Assume $\exists\, p, q$ coprime with $p^2 = 2q^2$

1. p^2 is even $\implies$ p is even (odd^2 is odd)
2. Write $p = 2r$. Then $4r^2 = 2q^2$, so $2r^2 = q^2$
3. q^2 is even $\implies$ q is even
4. Both p, q even $\implies$ contradiction with coprimality

Challenge:

- formalize this in LPTP down to Peano-level axioms
- evaluate LLM assistance
- compare with ATPs (automated theorem provers, ICLP 2025)

Formalization: Key Definitions

New symbols added to the specification language $\hat{\mathcal{L}}$:

$$
\begin{aligned}
divisor(x, y) &\leftrightarrow \exists z\, (nat^s(z) \wedge x * z = y) \\
coprime(p, q) &\leftrightarrow \forall r\, (nat^s(r) \wedge divisor(r, p) \wedge divisor(r, q) \to r = \mathrm{s}(0)) \\
square(n) = p &\leftrightarrow times^s(n, n, p) \qquad (\text{given } nat^s(n))
\end{aligned}
$$

- *square* requires proving *existence* and *uniqueness* of $n \times n$
- Reuses LPTP libraries: `nat`, `gcd`

Top-Down Proof Skeleton

Main theorem formalized in LPTP:

$$\forall p, q \, \big(nat^s(p) \wedge nat^s(q) \wedge coprime(p, q) \;\rightarrow\; \neg\, square(p) = 2 \cdot square(q)\big)$$

Methodology:

1. Write the full proof using `by gap` for unproven lemmas
2. LPTP validates the *logical structure* immediately
3. Eight auxiliary lemmas identified (e.g. `evenpp:evenp`, `simplify:by2`, . . .)
4. Then: fill the gaps with LLM help

Auxiliary Lemmas

Lemma	Statement (informal)	Difficulty
`nat:natsquare`	$nat(n) \to nat(square(n))$	easy
`twotimes:even`	$n = 2p \to even(n)$	easy
`even:twotimes`	$even(n) \to \exists p.\ n = 2p$	easy
`sqr2:4`	$square(2) = 4$	easy
`even:div2`	$even(n) \to divisor(2, n)$	easy
`npq:nnppqq`	$n = pq \to square(n) = square(p)square(q)$	medium
`simplify:by2`	$4n = 2p \to 2n = p$	medium
`evenpp:evenp`	$even(square(p)) \to even(p)$	**hard**

Dependency Graph of the Lemmas

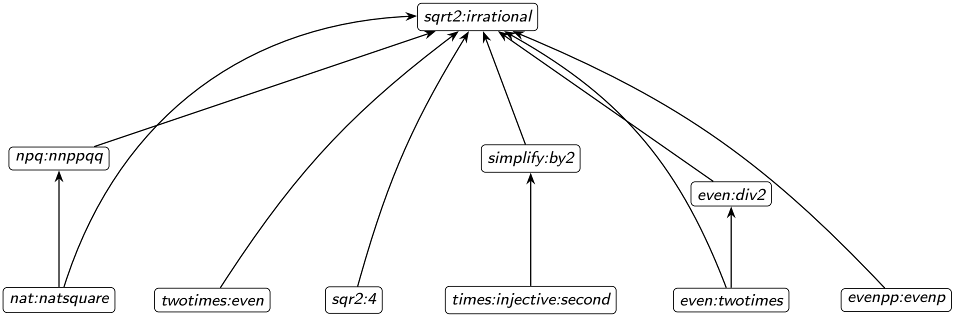
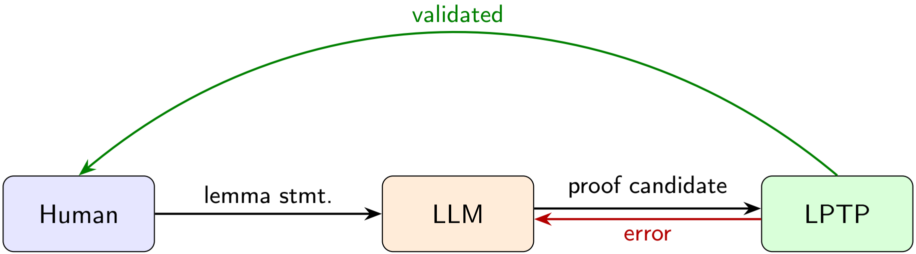


Figure: Dependency graph of the lemmas of the previous table. An arrow $A \to B$ means A is used in the proof of B.

Interaction with the LLM

- **Model:** Claude Opus 4.5 (Anthropic), Chat mode.
- **In-context learning:**
  - LPTP user manual (PDF)
  - Two LPTP libraries: Peano numbers + lists (code & proofs)
- **Workflow:** state each lemma in LPTP syntax → ask Claude to prove it → paste into LPTP → check



Results: First Round (Table 1)

Lemma	**Shorthand**	**LLM**	**ATP**
`nat:natsquare`	$nat(n) \to nat(square(n))$	DP	✓
`twotimes:even`	$2p = n \to even(n)$	DP	–
`even:twotimes`	$even(n) \to n = 2p$	DP	–
`sqr2:4`	$square(2) = 4$	DP	✓
`even:div2`	$even(n) \to divisor(2, n)$	DP	✓
`npq:nnppqq`	$n = pq \to square(n) = square(p)square(q)$	BF	–
`simplify:by2`	$4n = 2p \to 2n = p$	BF	–
`evenpp:evenp`	$even(square(p)) \to even(p)$	**NP**	–
`sqrt2:irrational`	main theorem	PG	✓

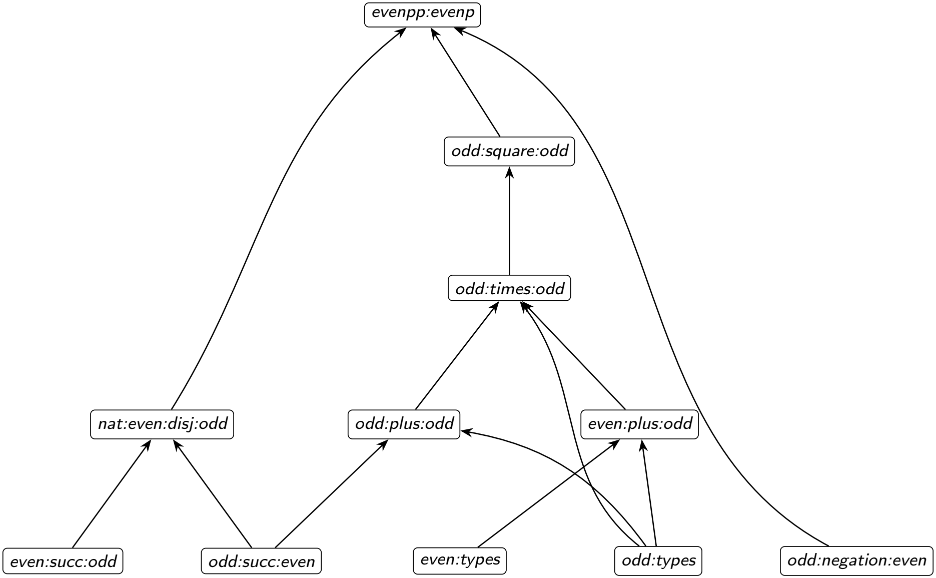
DP = directly proven, BF = back & forth, NP = not proven, PG = proof given.
ATP = automated theorem prover (E / Vampire, 20 s timeout)

The Hard Lemma: `evenpp:evenp`

$$nat(p) \wedge even(square(p)) \;\rightarrow\; even(p)$$

- Claude could *not* find this proof autonomously
- **Human intervention:** decompose into 15 auxiliary lemmas about parity:
  - $even(p) \rightarrow odd(p+1)$, $odd(p) \rightarrow even(p+1)$
  - $nat(p) \rightarrow even(p) \vee odd(p)$ (case split)
  - $odd(m) \wedge odd(n) \rightarrow odd(m \times n)$
  - $odd(p) \rightarrow odd(square(p))$, etc.
- With a **hint** (proof sketch), Claude generates the final proof $\Rightarrow$ LPTP validates

Dependency Graph: evenpp:evenp



Back to Prolog: Operational Consequence

- Define:

```
sqrt2_is_rational :- nat(S), plus(P,Q,S), ...
```

- Termination cannot be established $\Rightarrow$ we cannot prove *failure*
- Instead, prove $\neg$`sqrt2_is_rational`^{s}: the query *cannot succeed*

Bridging lemmas (Claude in Code mode, Opus 4.6):

Lemma	**LLM**	**ATP**
`divisor:of:one`	DP	✓
`gcd:one:coprime`	DP	–
`plus:double:twotimes`	DP	✓
`sqrt_two:irrational:query` ($\neg sqrt2\_is\_rational^s$)	DP	✓

Chat Mode vs. Code Mode

Chat Mode (Opus 4.5)

- Human copy/pastes between Claude and LPTP
- In-context learning: manual + 2 libraries
- Good for simple lemmas
- Back & forth on errors

Code Mode (Opus 4.6)

- Claude calls the LPTP proof checker directly via terminal
- Access to *full* LPTP source (7 kLOC)
- Deeper understanding of proof-checking mechanism
- Faster error correction

Key insight

The proof checker **eradicates** hallucinations
Each proposed proof is either *correct* or *wrong with a precise error location*

Related Work & Positioning

- **17 Provers of the World** (Wiedijk, 2006): $\sqrt{2}$ irrational in 17 proof assistants
  LPTP not included
- **LLM + Proof Assistants:**
  - DeepSeek-Prover-V1.5 (Lean 4, specialized model)
  - COPRA (GPT-4 + Rocq/Lean, backtracking search)
  - NLIR (GPT-4 + Rocq via Pétanque, natural language intermediate)
  - Hermes (informal reasoning → Lean translation)
- **ATP approach** (Mesnard et al. 2024/2026): plug FOL ATPs (E, Vampire) into LPTP; ~80% success rate on library
- **This work:** first LLM integration with LPTP; off-the-shelf model; proof-step generation

Conclusion & Future Work

Contributions:

- First interaction between LPTP and an LLM
- Semi-automated LPTP proof of the irrationality of $\sqrt{2}$
- Demonstrated both Chat and Code modes – Code mode more fluid
- Proof checker is essential to control LLM hallucinations

Future work:

- Evaluate LPTP/LLM on a broader set of Prolog verification problems

Thank you!

https://github.com/FredMesnard/LPTP-LLM-sqrt2