ALA: an Agentic Approach for Translating Mathematics into Code for Proof-Assistant

Patricio Gallardo, Maziar Raissi, **Ke Zhang**, **Sudhir Murthy**

Department of Mathematics, University of California, Riverside

UC RIVERSIDE Department of Mathematics

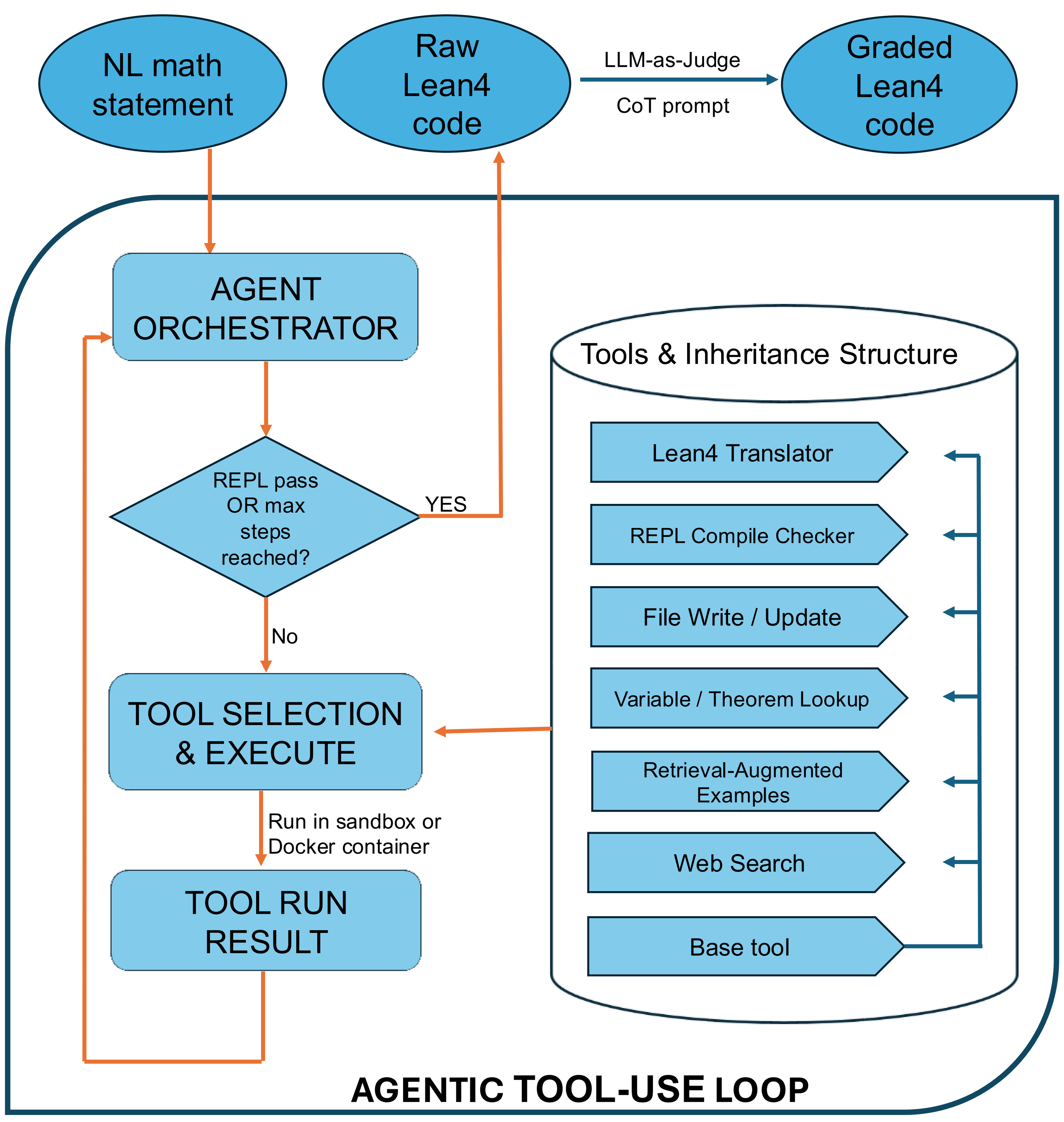
Introduction

Formalization is the activity of translating mathematics written in pen-paper into verifiable code. Our goal is to leverage AI to auto-formalize mathematical statements into faithful Lean4 code.

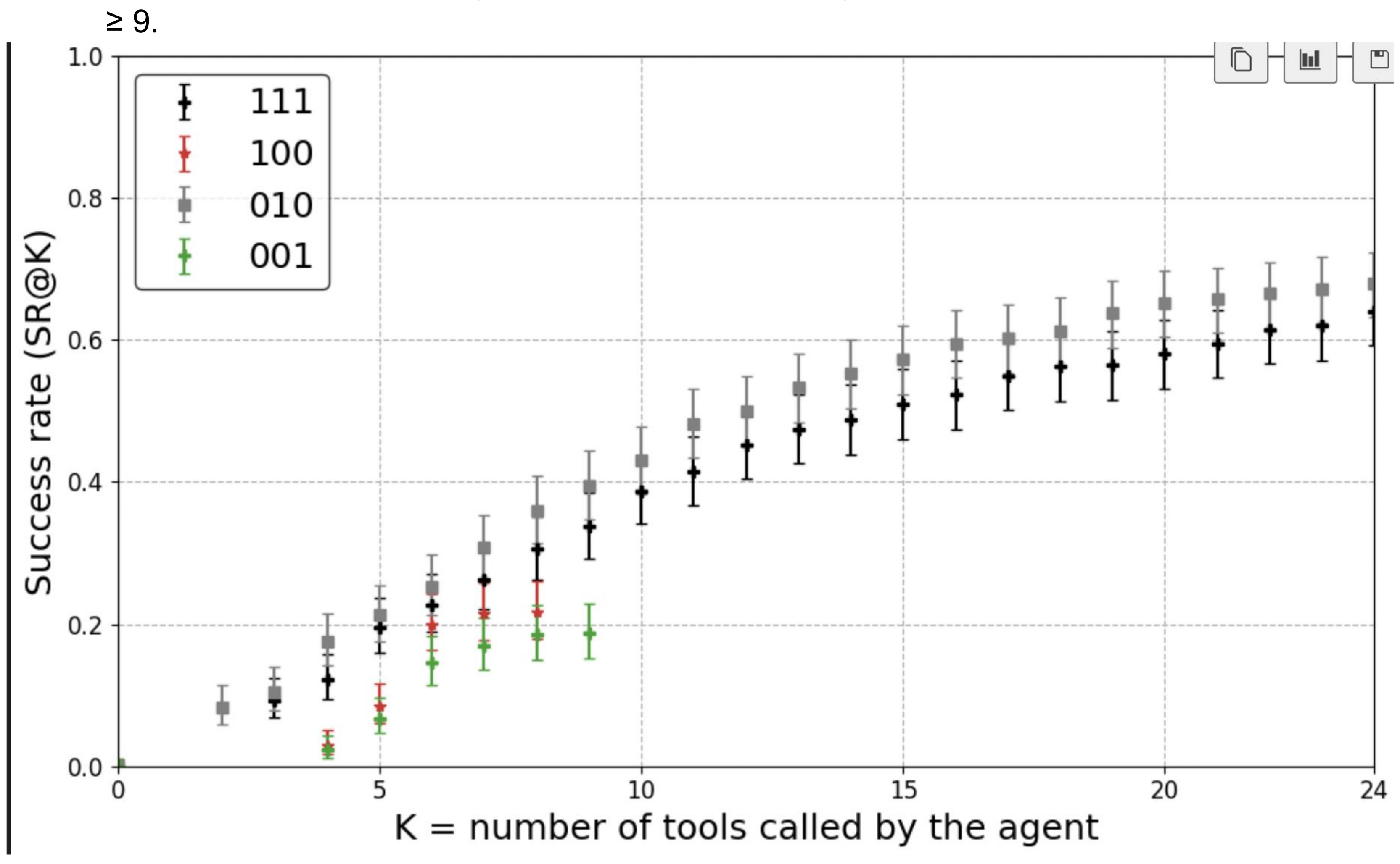
Challenges:

- LLM's hallucinate, applying false theorems and citing non-existent data types
- Limited datasets for non-Olympiad, upper-division and graduate level mathematics.
- Correct Lean4 code ≠ faithful translation. Need semantic validation.
- Traditional mathematics is based on Set Theory whereas Lean4 is based on Type Theory.

Our Approach



We introduce an agentic translation pipeline where an LLM iteratively refines Lean4 code using tool feedback. We employ an LLM-as-judge evaluation stage: a translation is accepted only if it compiles successfully and receives a semantic score ≥ 9.



We evaluate the agent under four tool configurations (via prompt-level toggling). Binary codes indicate which tool groups are active: 111 = all 6 tools enabled, 100 = translator only, 010 = feedback tools only (REPL + theorem checker), 001 = search tools only (RAG** + online search**).
After the agent completes its reasoning-and-tool-calling process, the resulting outputs are evaluated in a second stage by an LLM-as-judge.

Case Study

For each $\theta \in \mathbb{R}$, $H_\theta = H_0 R_{-2\theta}$, where

$$H_0 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \quad R_\phi = \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix}.$$

```
import Mathlib
open Real

def H0 : Matrix (Fin 2) (Fin 2) ℝ :=
 fun i j => if i = (0 : Fin 2) then
         if j = (0 : Fin 2) then 1 else 0
            else if j = (0 : Fin 2) then 0 else -1

noncomputable def R (ϕ : ℝ) : Matrix
(Fin 2) (Fin 2) ℝ := fun i j =>
   if i = (0:Fin 2) then
   if j = (0:Fin 2) then cos ϕ else -sin ϕ else if j = (0:Fin 2)
  then sin ϕ
   else cos ϕ

noncomputable def H (θ:ℝ): Matrix (Fin 2)
(Fin 2) ℝ := H0 * R (-2 * θ)

theorem H_theta_eq_H0_mul_R_neg_two_theta
(θ:ℝ) : H θ = H0 * R (-2 * θ) := by sorry
```

Key Contributions

- We introduce an agent with access to tools such as specialist LLM (Herald), to iteratively refine Lean4 code.
- We introduce a dataset of 400+ mathematical statements in analysis, algebra, and topology.
- Achieves a 3× improvement in equivalent translations vs. fine-tuned translator baseline (22.5% → ~70% best configuration).

Limitations & Future Work

- **No agent introspection**
  We did not trace why the agent chooses certain tools. Future work: add tool-use rationales or CoT logging if possible.
- **Raw REPL too verbose**
  Full REPL dumps may exceed small model context. Future: summarize or filter REPL but retain full output for tactics for large LLM.
- **Agent laziness**
  When feedback tool is not available, the agent being lazy. Future: encourage self-refinement through prompt engineering.

Get in Touch

LinkedIn

Full paper (PDF)

UC RIVERSIDE