DEPARTMENT OF MATHEMATICS · NOVEMBER 2026

Agentic AI for mathematical reasoning and verification

Rethlas, Danus, Pharos, and OSVerify

SHURUI LIU
Stanford University

Plan

I Agents
What are agents and harnesses?

II Our systems
Rethlas, Danus, Pharos, OSVerify

III How to use and contribute
Users' guide and how to contribute

IV Future work
Understanding, collaboration, creativity, evaluation

PART I

Agents and harnesses

What they are, what ChatGPT and Codex already provide, and what still goes wrong

What are agents and harnesses?

A definition from CS

Agent = **Model** + **Harness**

the trained weights

everything else: prompts, tools, memory, loops, orchestration

"If you're not the model, you're the harness." V. Trivedy, LangChain, 2026

What are agents and harnesses?

A definition from CS

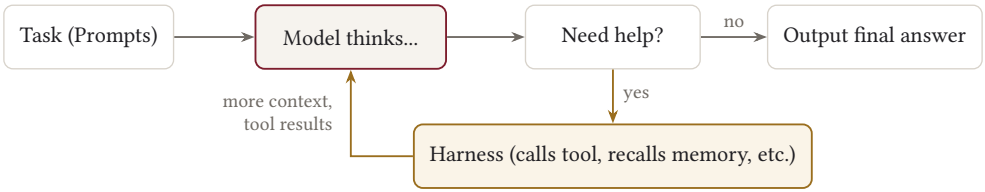


"If you're not the model, you're the harness." V. Trivedy, LangChain, 2026

A working definition

A harness is **context management**: it gives the model exactly the context it needs to solve a task.

Harness is Context Management



The model only sees its context. Each time it asks for help, the harness decides **what enters the context next**.

What a harness contains

Infrastructure
sandbox, workspace, run id

Loop support
compaction, keep going

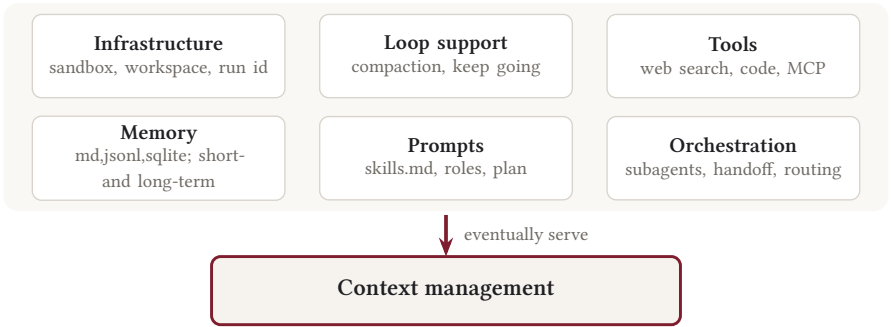
Tools
web search, code, MCP

Memory
md,jsonl,sqlite; short- and long-term

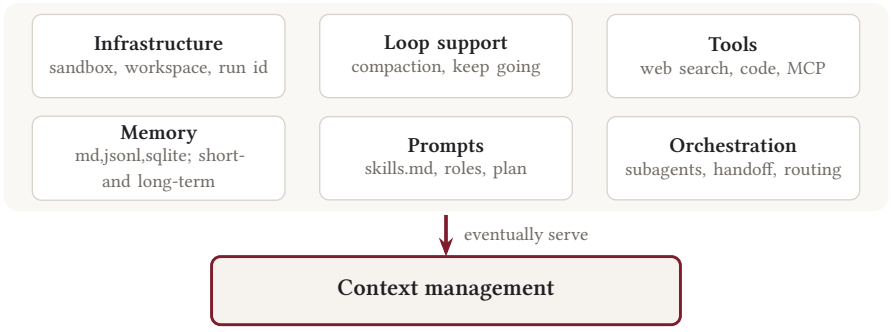
Prompts
skills.md, roles, plan

Orchestration
subagents, handoff, routing

What a harness contains



What a harness contains



Goal: give the LLM (a sentence-completion machine) the right context: everything useful for the task, and nothing irrelevant.

Harnesses you already used

Foundation model + **Harness**

the strongest models

general-purpose harnesses

ChatGPT Pro (web)
Extended thinking, web search, running code, etc.

Codex / Claude Code
An agent on your computer that can run shell commands, access your files, etc.

Built-in controls

`/goal`

Keep going until the goal is achieved.

Built-in controls

`/goal`

Keep going until the goal is achieved.

`max`

Deeper reasoning per step.

Built-in controls

`/goal`

Keep going until the goal is achieved.

`max`

Deeper reasoning per step.

`ultra`

More subagents in parallel.

Built-in controls

`/goal`

Keep going until the goal is achieved.

`max`

Deeper reasoning per step.

`ultra`

More subagents in parallel.

> However, these basic harnesses are for **general purposes**, not specifically for mathematics.

What still needs improvement?

1. Workflow for math research

Basic orchestration of multi-agents, procedures, etc., is not optimal for math research.

What still needs improvement?

1. Workflow for math research

Basic orchestration of multi-agents, procedures, etc., is not optimal for math research.

2. Context engineering

Irrelevant or misleading information should not enter the context.

What still needs improvement?

1. Workflow for math research

Basic orchestration of multi-agents, procedures, etc., is not optimal for math research.

2. Context engineering

Irrelevant or misleading information should not enter the context.

3. Long Horizon

With several lines of work in one context (topic A, then topic B), the model loses track of which one it is pursuing.

What still needs improvement?

1. Workflow for math research

Basic orchestration of multi-agents, procedures, etc., is not optimal for math research.

2. Context engineering

Irrelevant or misleading information should not enter the context.

3. Long Horizon

With several lines of work in one context (topic A, then topic B), the model loses track of which one it is pursuing.

Harness matters at least when the task exceeds the context window, or confuses the attention.

Why we work on harness?

1. CS reason: Training-free goes first

For most tasks, one should try training-free methods before training.

Why we work on harness?

1. CS reason: Training-free goes first

For most tasks, one should try training-free methods before training.

2. Practical reason: Training is too expensive in academia

Training a model comparable to frontier models on math capabilities is extremely hard.

Why we work on harness?

1. CS reason: Training-free goes first

For most tasks, one should try training-free methods before training.

2. Practical reason: Training is too expensive in academia

Training a model comparable to frontier models on math capabilities is extremely hard.

3. Other reasons: understanding AI

(1) Harness relies on understanding of the capabilities of base models, and (2) we get intuitions on how to improve base models during harness design.

Why we work on harness?

1. CS reason: Training-free goes first

For most tasks, one should try training-free methods before training.

2. Practical reason: Training is too expensive in academia

Training a model comparable to frontier models on math capabilities is extremely hard.

3. Other reasons: understanding AI

(1) Harness relies on understanding of the capabilities of base models, and (2) we get intuitions on how to improve base models during harness design.

Harness matters at least when the task exceeds the context window, or confuses the attention.

Working principles

First principles: Context Engineering

Provide the correct context!

Math intuitions: Mimic Human Workflow

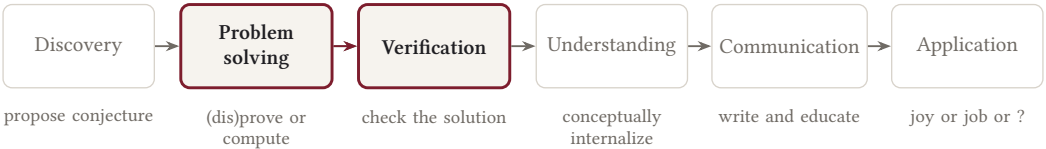
Mimic how human experts solve the task.

CS intuitions: scale the compute meaningfully

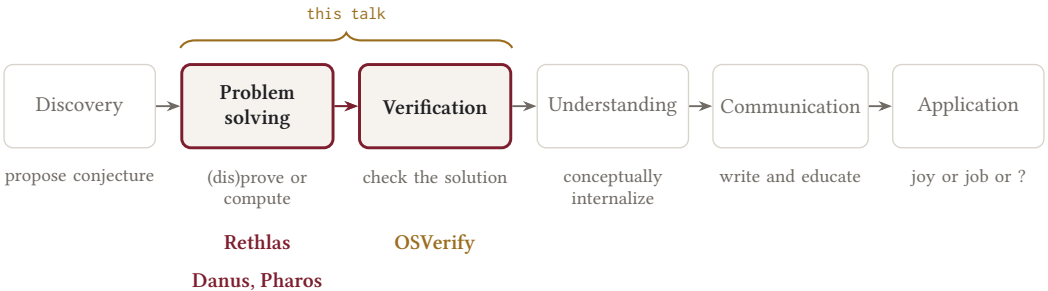
Scale the compute without too much waste, and compute can buy intelligence.

Harness matters at least when the task exceeds the context window, or confuses the attention.

Math research loop



Math research loop



Three systems

1 Rethlas

Lightweight: One agent proves, a fresh one referees, in a loop.

arXiv:2604.03789

2 Danus

Orchestration: A research group of agents sharing a DAG of verified facts.

arXiv:2607.06447

3 Pharos

Danus rebuilt for stronger models: planning, writing, working with experts.

frenzymath/Pharos

4 OSVerify

A stricter referee: one agent suspects errors, another agent confirms them.

arXiv:2610.18089

FOR EACH SYSTEM: RESULTS, THEN DESIGN (SHORT), THEN CASE STUDIES

PART II.1

Rethlas

A prover and a referee in a loop, persistently attacking one problem

RETHLAS H. Ju*, G. Gao*, J. Jiang*, B. Wu*, Z. Sun*, S. Liu*, …, L. Xiao, R. Liu, B. Dai, B. Dong

* *equal contribution*

Same model, with and without the harness

Problem	Subject	Reference	Model	Model alone	With Rethlas
Rational cocharacters	algebraic groups	§5.1.1, arXiv:2604.03789	GPT-5.5-Pro	✗ wrong proof	✓ 44 min
KL polynomials	matroids	arXiv:2607.24186	GPT-5.6 Sol	✗ incomplete	✓ 4 h 59 min
Automorphic commutator relation	relative geometric Langlands	arXiv:2610.05781	GPT-6-Astra	✗ incomplete	✓ 61 min

> In each case, only the statement was given.

Results across fields

Area	Results	System
Commutative algebra	Anderson's question; eight CFFG conjectures; Boij–Söderberg questions (Erman–Sam)	Rethlas
Algebraic groups	rational conjugacy of cocharacters	Rethlas
p-adic Hodge theory	refined lift-independence; a genus-separation counterexample	Rethlas
Analysis, probability	Brezis problems; injectivity of phase retrieval	Rethlas
Combinatorics	matroid KL polynomials are not real-rooted	Rethlas
Algebraic Number theory, Geometric Langlands	Manin constant; automorphic commutator relation	Rethlas + OSVerify

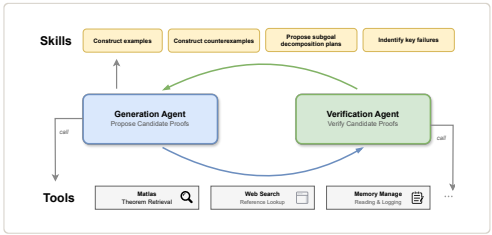
frenzymath/

Rethlas_results

A partial, growing list, checked by human experts before release.

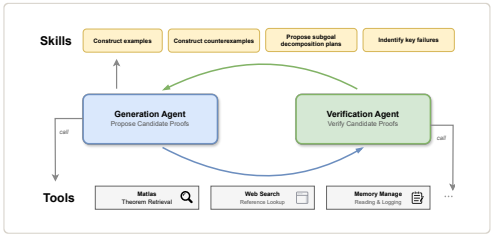
Rethlas in one slide

- A **generation agent** writes an informal proof *blueprint*, with skills (habits), a search engine (the library), and a notebook (memory).



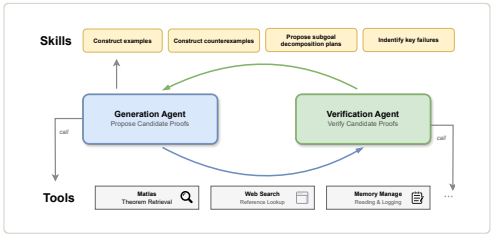
Rethlas in one slide

- A **generation agent** writes an informal proof *blueprint*, with skills (habits), a search engine (the library), and a notebook (memory).
- A **verification agent** referees it: a fresh process per submission returns a verdict and repair hints.



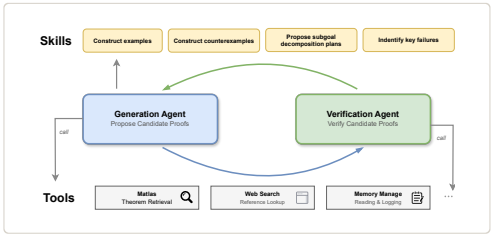
Rethlas in one slide

- A **generation agent** writes an informal proof *blueprint*, with skills (habits), a search engine (the library), and a notebook (memory).
- A **verification agent** referees it: a fresh process per submission returns a verdict and repair hints.
- Repeat until acceptance: zero critical errors *and* zero gaps.



Rethlas in one slide

- A **generation agent** writes an informal proof *blueprint*, with skills (habits), a search engine (the library), and a notebook (memory).
- A **verification agent** referees it: a fresh process per submission returns a verdict and repair hints.
- Repeat until acceptance: zero critical errors *and* zero gaps.



> Human workflow: **The author is not the referee**.

Habits, the library, the notebook

Skills = habits

Toy examples, counterexamples, literature search, decomposition, direct and recursive proving, identifying the key failure.

Loop: *assess, choose a skill, act.*

Habits, the library, the notebook

Skills = habits

Toy examples, counterexamples, literature search, decomposition, direct and recursive proving, identifying the key failure.

Loop: *assess, choose a skill, act.*

Tool = External resources (e.g. the library)

Matlas (`matlas.ai`): semantic search over ∼13.6M theorem statements from arXiv.

Think first, then look it up.

Habits, the library, the notebook

Skills = habits

Toy examples, counterexamples, literature search, decomposition, direct and recursive proving, identifying the key failure.

Loop: *assess, choose a skill, act.*

Tool = External resources (e.g. the library)

Matlas (`matlas.ai`): semantic search over ∼13.6M theorem statements from arXiv.

Think first, then look it up.

Memory = the notebook

Toy examples, failed paths, partial results, kept as structured notes.

Habits, the library, the notebook

Skills = habits

Toy examples, counterexamples, literature search, decomposition, direct and recursive proving, identifying the key failure.

Loop: *assess, choose a skill, act.*

Tool = External resources (e.g. the library)

Matlas (`matlas.ai`): semantic search over ∼13.6M theorem statements from arXiv.

Think first, then look it up.

Memory = the notebook

Toy examples, failed paths, partial results, kept as structured notes.

Failed paths are mandatory

Every dead end stays on record.

Case Study 1: reference/new concept?

QUESTION (SUGGESTED BY B. CONRAD) G a connected smooth affine group over a field k, $\mathbb{X}_*(G) = \mathrm{Hom}_{k\text{-gp}}(\mathrm{GL}_1, G)$. For K/k, is $\mathbb{X}_*(G)/G(k) \to \mathbb{X}_*(G_K)/G(K)$ injective?

Known for reductive G, or perfect k.

Case Study 1: reference/new concept?

QUESTION (SUGGESTED BY B. CONRAD) G a connected smooth affine group over a field k, $\mathbb{X}_*(G) = \mathrm{Hom}_{k\text{-gp}}(\mathrm{GL}_1, G)$. For K/k, is $\mathbb{X}_*(G)/G(k) \to \mathbb{X}_*(G_K)/G(K)$ injective?

Known for reductive G, or perfect k.

✖ GPT-5.5 Pro (web): a wrong proof, no meaningful progress.

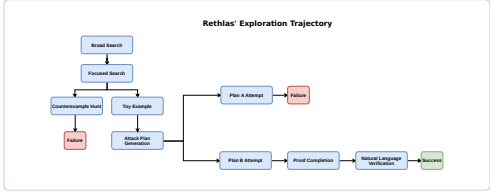
Case Study 1: reference/new concept?

QUESTION (SUGGESTED BY B. CONRAD) G a connected smooth affine group over a field k, $\mathbb{X}_*(G) = \mathrm{Hom}_{k\text{-gp}}(\mathrm{GL}_1, G)$. For K/k, is $\mathbb{X}_*(G)/G(k) \to \mathbb{X}_*(G_K)/G(K)$ injective?

Known for reductive G, or perfect k.

✗ GPT-5.5 Pro (web): a wrong proof, no meaningful progress.

✓ Rethlas (GPT-5.5): a correct proof in 44 minutes, through the "quasi-reductive" case (AI coined "wound unipotent case").



Case Study 1: reference/new concept?

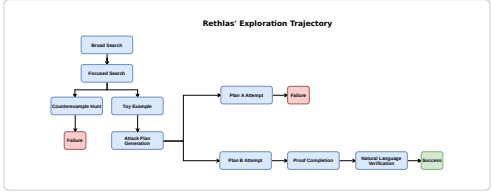
QUESTION (SUGGESTED BY B. CONRAD) G a connected smooth affine group over a field k, $\mathbb{X}_*(G) = \mathrm{Hom}_{k\text{-gp}}(\mathrm{GL}_1, G)$. For K/k, is $\mathbb{X}_*(G)/G(k) \to \mathbb{X}_*(G_K)/G(K)$ injective?

Known for reductive G, or perfect k.

✗ GPT-5.5 Pro (web): a wrong proof, no meaningful progress.

✓ Rethlas (GPT-5.5): a correct proof in 44 minutes, through the "quasi-reductive" case (AI coined "wound unipotent case").

- One "mistake": a **citation substitution** (due to copyright issue). The needed result is in a book the agent could not retrieve, so it cited a survey.



Case 2: Extract the method?

CONJECTURES The Kazhdan–Lusztig polynomial $P_M(t)$ of a matroid M is real-rooted (Gedeon–Proudfoot–Young, 2017) and log-concave (Elias–Proudfoot–Wakefield, 2016). Evidence from uniform, q-niform, fan, wheel, whirl matroids.

Prompt: the definition of P_M, and "prove or disprove that all zeros of $P_M(t)$ lie on the negative real axis."

R. Cheng, S. Liu, Kazhdan–Lusztig polynomials of matroids need not be unimodal, arXiv:2607.24186.

Case 2: Extract the method?

CONJECTURES The Kazhdan–Lusztig polynomial $P_M(t)$ of a matroid M is real-rooted (Gedeon–Proudfoot–Young, 2017) and log-concave (Elias–Proudfoot–Wakefield, 2016). Evidence from uniform, q-niform, fan, wheel, whirl matroids.

Prompt: the definition of P_M, and "prove or disprove that all zeros of $P_M(t)$ lie on the negative real axis."

✔ Rethlas (GPT-5.6 Sol): explicit counterexample in 4 h 59 min, by deleting points from a finite projective geometry.

R. Cheng, S. Liu, Kazhdan–Lusztig polynomials of matroids need not be unimodal, arXiv:2607.24186.

Case 2: Extract the method?

CONJECTURES The Kazhdan–Lusztig polynomial $P_M(t)$ of a matroid M is real-rooted (Gedeon–Proudfoot–Young, 2017) and log-concave (Elias–Proudfoot–Wakefield, 2016). Evidence from uniform, q-niform, fan, wheel, whirl matroids.

Prompt: the definition of P_M, and "prove or disprove that all zeros of $P_M(t)$ lie on the negative real axis."

✔ Rethlas (GPT-5.6 Sol): explicit counterexample in 4 h 59 min, by deleting points from a finite projective geometry.

✖ GPT-5.6 Sol Ultra (web), two prompts: incomplete.

R. Cheng, S. Liu, Kazhdan–Lusztig polynomials of matroids need not be unimodal, arXiv:2607.24186.

Case 2: Extract the method?

CONJECTURES The Kazhdan–Lusztig polynomial $P_M(t)$ of a matroid M is real-rooted (Gedeon–Proudfoot–Young, 2017) and log-concave (Elias–Proudfoot–Wakefield, 2016). Evidence from uniform, q-niform, fan, wheel, whirl matroids.

Prompt: the definition of P_M, and "prove or disprove that all zeros of $P_M(t)$ lie on the negative real axis."

✔ Rethlas (GPT-5.6 Sol): explicit counterexample in 4 h 59 min, by deleting points from a finite projective geometry.

✖ GPT-5.6 Sol Ultra (web), two prompts: incomplete.

Humans understood the proof

Human turned the construction into a method, gave a geometric interpretation, and proved that P_M need not be **unimodal**.

R. Cheng, S. Liu, Kazhdan–Lusztig polynomials of matroids need not be unimodal, arXiv:2607.24186.

Case 2: Extract the method?

CONJECTURES The Kazhdan–Lusztig polynomial $P_M(t)$ of a matroid M is real-rooted (Gedeon–Proudfoot–Young, 2017) and log-concave (Elias–Proudfoot–Wakefield, 2016). Evidence from uniform, q-niform, fan, wheel, whirl matroids.

Prompt: the definition of P_M, and "prove or disprove that all zeros of $P_M(t)$ lie on the negative real axis."

✔ Rethlas (GPT-5.6 Sol): explicit counterexample in 4 h 59 min, by deleting points from a finite projective geometry.

✖ GPT-5.6 Sol Ultra (web), two prompts: incomplete.

Humans understood the proof

Human turned the construction into a method, gave a geometric interpretation, and proved that P_M need not be **unimodal**.

AI only did minimal to solve the task prompted

It did not generalize the construction, explain it geometrically, or see that it disproves the second conjecture.

R. Cheng, S. Liu, Kazhdan–Lusztig polynomials of matroids need not be unimodal, arXiv:2607.24186.

Why generate-verify loop is not enough

No decomposition of the problem

A mathematician accumulates many small facts and adjusts strategy.

Why generate-verify loop is not enough

No decomposition of the problem

A mathematician accumulates many small facts and adjusts strategy.

Linear logical structure

A problem may need several people on different routes and different routes interact; the logic structure may not be linear.

Why generate-verify loop is not enough

No decomposition of the problem

A mathematician accumulates many small facts and adjusts strategy.

Linear logical structure

A problem may need several people on different routes and different routes interact; the logic structure may not be linear.

Context engineering

Every round the generator rereads the whole draft with all proofs, even when only the statements are needed and should be used as blackbox.

Why generate-verify loop is not enough

No decomposition of the problem

A mathematician accumulates many small facts and adjusts strategy.

Linear logical structure

A problem may need several people on different routes and different routes interact; the logic structure may not be linear.

Context engineering

Every round the generator rereads the whole draft with all proofs, even when only the statements are needed and should be used as blackbox.

We want to **scale compute** on generation and verification: from a single author-referee loop to a research group and meta-verifier.

PART II.2

Danus and Pharos

A research group of agents over a graph of verified lemmas; scale compute on generation

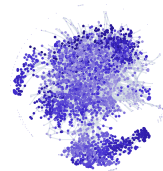
DANUS J. Liu, G. Gao, Z. Sun, B. Wu, S. Liu, …, B. Dong

PHAROS G. Gao*, Z. Sun*, B. Wu*, S. Liu*, J. Jiang, B. Dong

* *equal contribution*

Tangent classes of matroids

PROBLEM (MAIN RESULTS OF A PAPER BY R. CHENG)
Construct a tangent class $T_{M,\mathcal{G}}$ in the combinatorial K-ring of any loopless matroid, extending the tangent bundle of the wonderful compactification, with HRR and Chern-number properties. Cheng's own preprint was withheld from all systems.



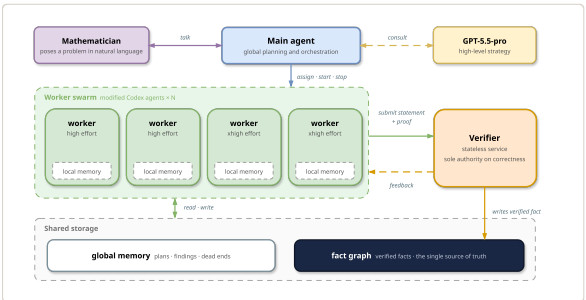
Same verbatim prompt		Outcome
GPT-5.5-pro (web)	model	✗ no solution
Rethlas, 3 runs	loop	✗ 0/3 passed the verifier agent
Danus	research group	✓ verified solution

3,157
verified facts

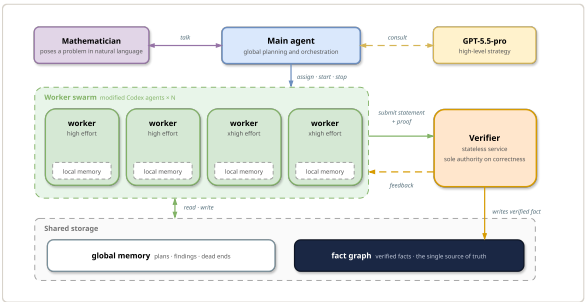
54
deepest chain

R. Cheng, S. Liu, G. Gao, Tangent classes of matroids and wonderful compactifications, arXiv:2607.05835.

A research group of agents

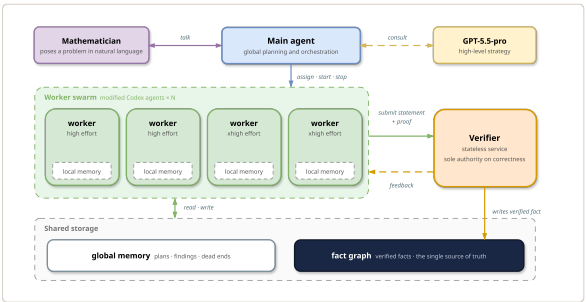


A research group of agents



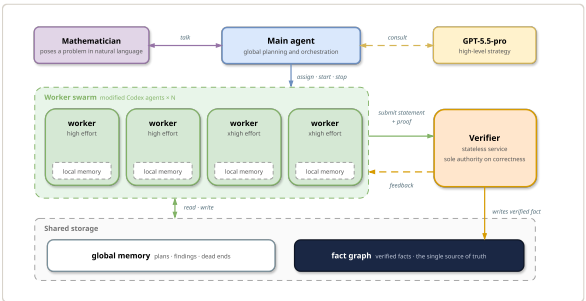
- **Main agent** = group leader: plans, assigns, monitors, reports. Does no mathematics itself.

A research group of agents



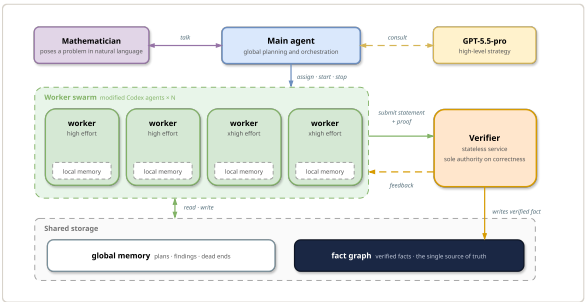
- **Main agent** = group leader: plans, assigns, monitors, reports. Does no mathematics itself.
- **Workers**: Rethlas provers, one lemma at a time.

A research group of agents



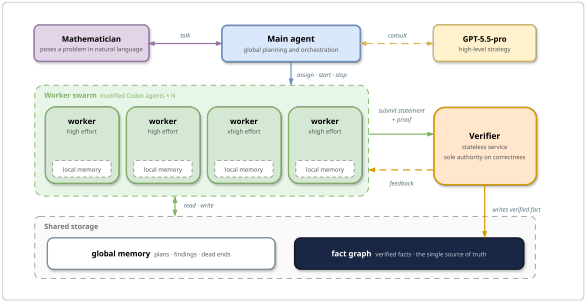
- **Main agent** = group leader: plans, assigns, monitors, reports. Does no mathematics itself.
- **Workers**: Rethlas provers, one lemma at a time.
- **Referee**: fresh for each lemma; the only way into the fact graph.

A research group of agents



- **Main agent** = group leader: plans, assigns, monitors, reports. Does no mathematics itself.
- **Workers**: Rethlas provers, one lemma at a time.
- **Referee**: fresh for each lemma; the only way into the fact graph.
- **Fact graph**: verified lemmas (statement + checked proof) and their dependencies, a DAG.

A research group of agents



- **Main agent** = group leader: plans, assigns, monitors, reports. Does no mathematics itself.
- **Workers**: Rethlas provers, one lemma at a time.
- **Referee**: fresh for each lemma; the only way into the fact graph.
- **Fact graph**: verified lemmas (statement + checked proof) and their dependencies, a DAG.

Engineering aspect: Three-tier memory management; facts graph search/submit MCP; paper-writing loop, etc.

Tangent classes: inside the run

- **7 workers**; ∼5 days, ∼50 h active.

Tangent classes: inside the run

- **7 workers**; ∼5 days, ∼50 h active.
- **3,157 verified facts**, 8,616 dependencies; chains up to **54** deep; **499** facts needed for the theorem.

Tangent classes: inside the run

- **7 workers**; ∼5 days, ∼50 h active.
- **3,157 verified facts**, 8,616 dependencies; chains up to **54** deep; **499** facts needed for the theorem.
- Shared notes: 636 proof attempts, 151 counterexamples, 25 dead ends.

Tangent classes: inside the run

- **7 workers**; ∼5 days, ∼50 h active.
- **3,157 verified facts**, 8,616 dependencies; chains up to **54** deep; **499** facts needed for the theorem.
- Shared notes: 636 proof attempts, 151 counterexamples, 25 dead ends.
- **Why Rethlas failed**: it conflated nearby notions (realizable matroid, wonderful model, toric variety; integral vs. rational K-theory).
  Indication: math subtlety can be overcomed by scaling compute (thinking longer and more carefully).

Tangent classes: inside the run

- **7 workers**; ∼5 days, ∼50 h active.
- **3,157 verified facts**, 8,616 dependencies; chains up to **54** deep; **499** facts needed for the theorem.
- Shared notes: 636 proof attempts, 151 counterexamples, 25 dead ends.
- **Why Rethlas failed**: it conflated nearby notions (realizable matroid, wonderful model, toric variety; integral vs. rational K-theory).
  Indication: math subtlety can be overcomed by scaling compute (thinking longer and more carefully).

More exploration

About 6× more facts were proved than the final proof uses.

Tangent classes: inside the run

- **7 workers**; ∼5 days, ∼50 h active.
- **3,157 verified facts**, 8,616 dependencies; chains up to **54** deep; **499** facts needed for the theorem.
- Shared notes: 636 proof attempts, 151 counterexamples, 25 dead ends.
- **Why Rethlas failed**: it conflated nearby notions (realizable matroid, wonderful model, toric variety; integral vs. rational K-theory).
  Indication: math subtlety can be overcomed by scaling compute (thinking longer and more carefully).

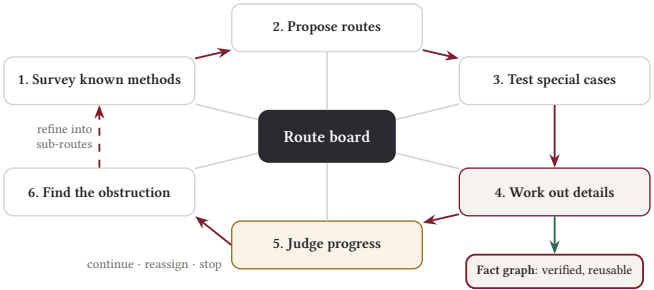
More exploration

About 6× more facts were proved than the final proof uses.

Scale compute

Scales the width and depth of the exploration by agents.

Pharos: a route board for strategy/planning



Known routes first

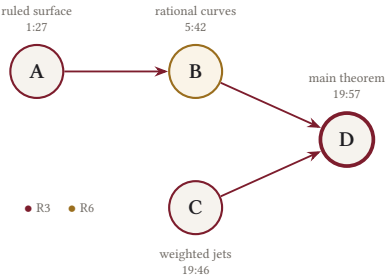
Techniques from the literature are tried first; removed consulting GPT-pro.

Also new in Pharos

Deeper lemmas, helpers, computation, expert hints.

An algebraic proof of the Miyaoka–Mori criterion

QUESTION (DU–GUO–XIE, APPENDIX F) For smooth projective X in characteristic zero and a curve $f\colon C \to X$, does $-K_X \cdot f_*[C] > 0$ alone give a rational curve through every point of $f(C)$, under a strict degree bound?



- **19 h 57 min**, no human intervention.
- **273** verified facts, only **4** in the proof; facts reused across routes.
- 7 routes, refined into 30 sub-routes.
- With expert hints at launch: 11 h 38 min, same proof.
- >300k lines of Lean 4 within a week.

B. Dong, G. Gao, B. Guo, Z. Sun, B. Wu, S.-Y. Xie, arXiv:2609.28465, Appendix B.

PART II.3

OSVerify

A stricter referee: scale compute on verification

OSVERIFY F. Nie*, H. Ye*, S. Liu, Z. Wang, S. Ermon, J. Zou

**equal contribution*

OSVerify as a referee

OpenAI preprints

11 recent OpenAI preprints on central problems (Kakeya in dimension 4, planar unit distances, …). Mistakes found in each, overlooked by OpenAI's internal review.

Manin Constant Conjecture

12 independent rounds of Opus 5.5, Fable 5.1, GPT-Astra-6 check found no error in an early draft. OSVerify with GPT-5.6-Sol reported one mistake, which affected three places in the paper.

Kamnitzer's ICM 2022 conjecture

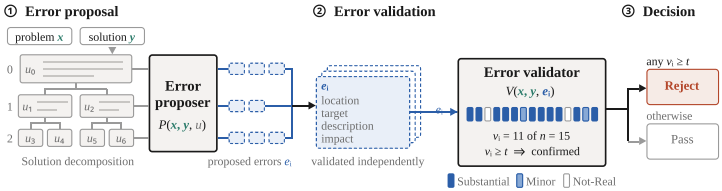
AI-assisted paper by Q. Tang (hexagon:2611.00006v1): Mirković–Vilonen and theta bases coincide.

Round 1: GPT-6-Pro drafted, Codex reviewed; OSVerify caught a major error.

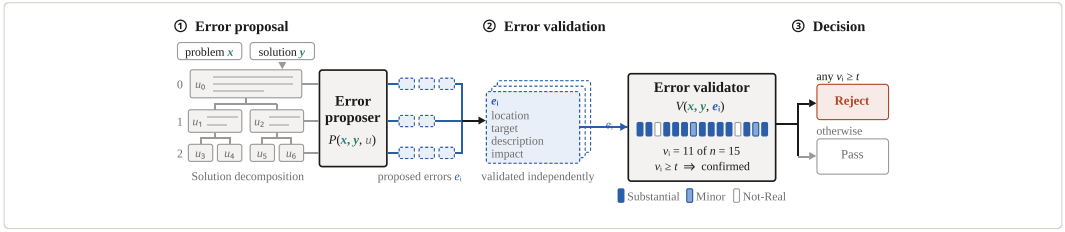
Round 2: GPT-6-Pro repaired; three Codex reviews found nothing; OSVerify caught another proof-breaking error.

Nie–Ye–Liu–Wang–Ermon–Zou, arXiv:2610.18089.

Finding an error vs. confirming one



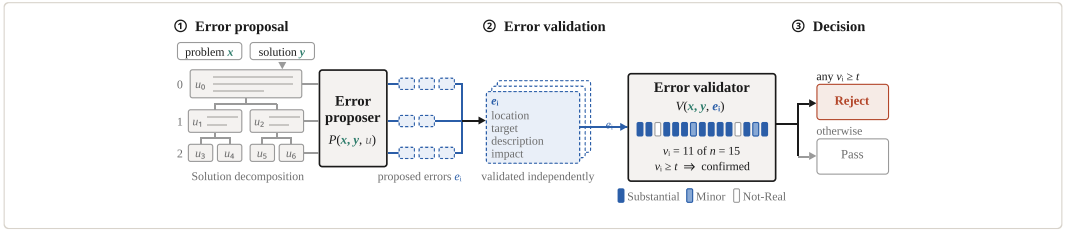
Finding an error vs. confirming one



Proposer goal: coverage

Break the proof into pieces at every scale; list **every** step that might be wrong.

Finding an error vs. confirming one



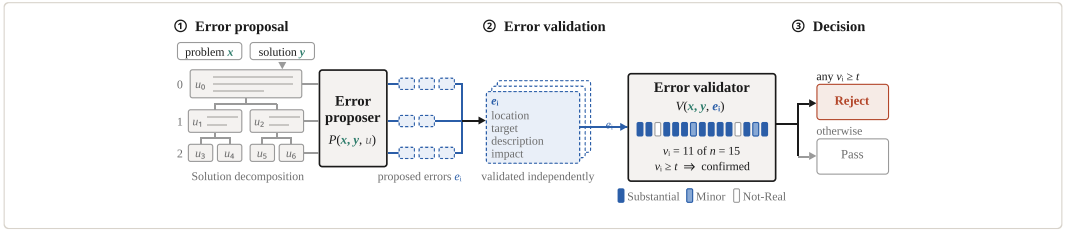
Proposer goal: coverage

Break the proof into pieces at every scale; list **every** step that might be wrong.

Validator goal: no false alarms

For each suspected error, decide: SUBSTANTIAL, MINOR, or NOT REAL.

Finding an error vs. confirming one



Proposer goal: coverage

Break the proof into pieces at every scale; list **every** step that might be wrong.

Validator goal: no false alarms

For each suspected error, decide: SUBSTANTIAL, MINOR, or NOT REAL.

Human workflow: a referee first *suspects* a step, then *checks* it; OSVerify uses a separate agent for each.

The Manin constant (Algebraic Number Theory)

THEOREM (CHI–LIU–NIE, ARXIV:2610.25672) The Manin constant conjecture is true; Stevens's Conjectures I and II are true.

Base models, expert prompting

ChatGPT-6 Pro, Claude Fable 5.1, Codex 6-Astra first proved only $c_\varphi \mid \deg \varphi$. Reaching $c_E \mid 2$, then $c_E = 1$, took many rounds of prompts from the authors.

Review

12 independent AI checks passed. OSVerify found one mistake: the claim $L_E \in \mathbb{Q}[[w]]$ for every $E/\mathbb{Q}_p$. It suggested a fix for the three affected places.

Harness

Rethlas (GPT-6 Astra) with OSVerify, given only the statement, reproduced essentially the same proof, rejecting ineffective strategies automatically.

Base model vs. harness

Problem	Model	Model alone	With harness
Cocharacters	GPT-5.5	✗ wrong proof	✓ Rethlas: correct, 44 min
KL polynomials	GPT-5.6 Sol	✗ incomplete	✓ Rethlas: counterexample, 5 h
Automorphic commutator	GPT-6 Astra	✗ incomplete	✓ Rethlas: proof, 1 h
Matroid tangent classes	GPT-5.5	✗ no solution	✓ Danus: solved (Rethlas 0/3)
Manin, solving	GPT-6 Astra	✗ many expert prompts	✓ Rethlas + OSVerify: statement only
Manin, review	GPT-6 Astra	✗ 12 AI checks passed	✓ OSVerify: one error found

The models are the same or comparable. What differs is the harness.

PART III

How to Use and Contribute

From Math perspective

How to use them: don't use a sledgehammer to crack a nut

System	Use it for	Stable runtime	Code (`https://github.com/`)
GPT-Pro (Web)	discussion or small questions	-	-
Codex (/goal)	a solvable problem	30min-3hr	-
Rethlas	solution might have simple logical structure	1-5hr	`frenzymath/Rethlas`
Pharos	a complicated problem	≈ weekly Pro quota (per 12hr)	`frenzymath/Pharos`
OSVerify	checking a paper or proof	1-5hr (<10% weekly quota)	`OSVerify`
Rethlas + OSV	Stricter verifier	1-5hr (<20% weekly quota)	`OSVerify/Rethlas-OSV`

You can ask Codex/Claude Code to help you download and run from source code. Rethlas-OSV has a desktop app. (Check out srliu.com/blog for detailed guidance)

Where mathematical expertise contributes

Human experience provides intuition for harness design

Distill how one does math and then design skills and orchestrations accordingly.

Where mathematical expertise contributes

Human experience provides intuition for harness design

Distill how one does math and then design skills and orchestrations accordingly.

Construct dataset and benchmark

This is important for evaluation of agentic systems and for model training.

Where mathematical expertise contributes

Human experience provides intuition for harness design

Distill how one does math and then design skills and orchestrations accordingly.

Find failure mode

Try a system and find how and why the system fails. This motivates new designs and helps debugging.

Construct dataset and benchmark

This is important for evaluation of agentic systems and for model training.

Where mathematical expertise contributes

Human experience provides intuition for harness design

Distill how one does math and then design skills and orchestrations accordingly.

Construct dataset and benchmark

This is important for evaluation of agentic systems and for model training.

Find failure mode

Try a system and find how and why the system fails. This motivates new designs and helps debugging.

Find new applications

New ways of using AI meaningfully to contribute to math.

PART IV

Future work

Meta-harnesses, understanding, and creativity

Open directions

From verification to understanding and writing

How can AI help humans understand papers? Clearer papers, better writing?

Human–AI collaboration

More effective human–AI collaboration in Danus, Pharos, …

Creativity and theory building

How to go beyond the “convex closure” (not well-defined!)?

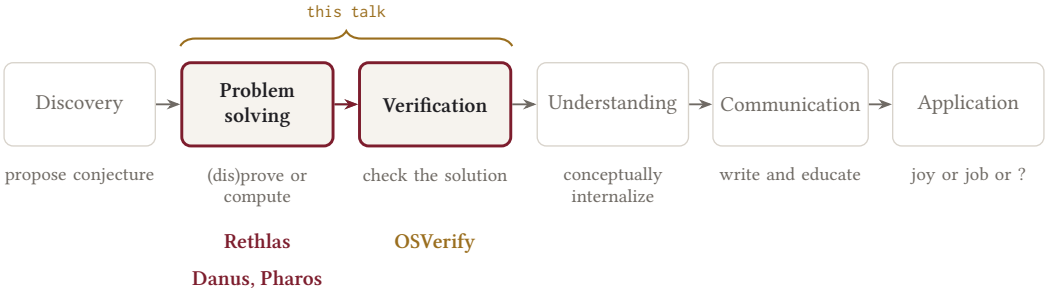
Evaluation

How do we benchmark research-level systems rigorously?

Self-improvement

Harnesses that improve themselves.

Summary



Design philosophy: 1. mimic human workflow, 2. context management, 3. scale test-time compute.

References

SYSTEMS

H. Ju, G. Gao, J. Jiang, B. Wu, Z. Sun, S. Liu, et al. *Automated Conjecture Resolution with Formal Verification.* arXiv:2604.03789.

J. Liu, G. Gao, Z. Sun, B. Wu, S. Liu, et al. *Danus: Orchestrating Mathematical Reasoning Agents with Fact-Graph Memory.* arXiv:2607.06447.

F. Nie, H. Ye, S. Liu, Z. Wang, S. Ermon, J. Zou. *One-Sided Verification Under Limited Expert Review.* arXiv:2610.18089.

G. Gao, Z. Sun, B. Wu, S. Liu, J. Jiang, B. Dong. *Pharos.* Appendix B of arXiv:2609.28465.

G. Gao. *Does the harness still matter?* Talk, November 2026.

Code: `github.com/frenzymath/{Rethlas, Danus, Archon, Pharos}`; `github.com/OSVerify`.
Problems and sources for all Rethlas results: `github.com/frenzymath/Rethlas_results`

MATHEMATICS

R. Cheng, S. Liu, G. Gao. *Tangent classes of matroids and wonderful compactifications.* arXiv:2607.05835.

R. Cheng, S. Liu. *Kazhdan–Lusztig polynomials of matroids need not be unimodal.* arXiv:2607.24186.

B. Dong, G. Gao, B. Guo, Z. Sun, B. Wu, S.-Y. Xie. *Constructing rational curves via jets on projective varieties with non-nef canonical bundle.* arXiv:2609.28465.

Z. Chi, S. Liu, P. Nie. *On the Manin constant.* arXiv:2610.25672.

S. Liu. *Automorphic commutator relations in relative geometric Langlands.* arXiv:2610.05781.

Q. Tang. *Equality of the Mirković–Vilonen and theta bases.* Hexagon preprint hexagon:2611.00006v1, 2026.

PROBLEMS

D. D. Anderson. Question on quasi-complete Noetherian local rings, 2014; see arXiv:2604.03789 for the statement and references.

Thank you

SHURUI LIU

Stanford University

github.com/frenzymath · matlas.ai