

Solid Foundations of Distributed Computing in the AI Era

Jukka Suomela
Aalto University

Intertwined threads

Three parallel developments:

1. Need for **robust foundations** of distributed computing theory
2. Modern **AI tools** can do TCS research
3. Rapid development of **formalization of mathematics**

Rapidly advancing area

- **The talk abstract that I wrote in May did not age well**
 - the use of AI in TCS research has changed from curiosity to *mainstream*
 - automatic formalization is no longer just possible, it is *quick and easy*
- **But the need for solid foundations is there, more pressing than ever...**

Personal perspective

- I will talk about my **personal** reflections and experiences on these topics
- I am looking at things especially from the perspective of **network algorithms**
 - distributed graph algorithms
 - **LOCAL model**
 - locally checkable labeling problems (LCLs)...

**Need for
robust
foundations**

Increasing importance

- Many **new connections** emerging between network algorithms and other fields
 - *physics*: quantum information theory
 - *mathematics*: stochastic processes, descriptive combinatorics
 - *TCS*: online and dynamic graph algorithms...
- Network algorithms serve as a **bridge** between fields — is it solid?

Our standard models

- There are hundreds of papers about **"the LOCAL model"**
- Few of them **agree** on all details
- Few of them even **specify** all details!
- Yet we tend to pretend minor details can be ignored, and everything still fits together...

Model variants

- **Some LOCAL model variants:**

- do we know n and Δ ? precisely? some bounds?
- what is the range?
- how much randomness?
 - can nodes pick a random neighbor?
 - can we communicate with neighbors?
- are state transitions synchronous/asynchronous?
- Monte Carlo or Las Vegas?
 - in expectation or with high probability?

Which of these really matter?

Can we ignore all these if we only care about “reasonable” problems?

(e.g. locally checkable problems)

Our project (2024–)

- **Precisely defined** models of distributed computing, with all assumptions and parameters **explicit**
- Understanding which theorems really hold for which **range** of models, in exactly what form
- Computer-readable **databases** of results, with computer-checkable proofs...

Surprises along the way

- Many details that seem to be largely ignored: can we **prove** that they are indeed irrelevant?
- *Shared randomness* shouldn't help with local problems? Wrong, it matters [\[ICALP 2025\]](#)
- *Non-computability* shouldn't help with local problems? Wrong, it matters [\[DISC 2026\]](#)

Surprises along the way

- We also tried to find **more restricted definitions** of “local problems” that would exclude many pathological examples
- It turned out that ***no such thing exists!***
- All reasonable definitions of “local problems” are essentially equally expressive [DISC 2026]

Robust foundations

- **Huge project**

- along the way, we will need to re-prove all key results in this area, in a formally precise manner, across all possible model variants...

- But now we have **new tools** available!

AI

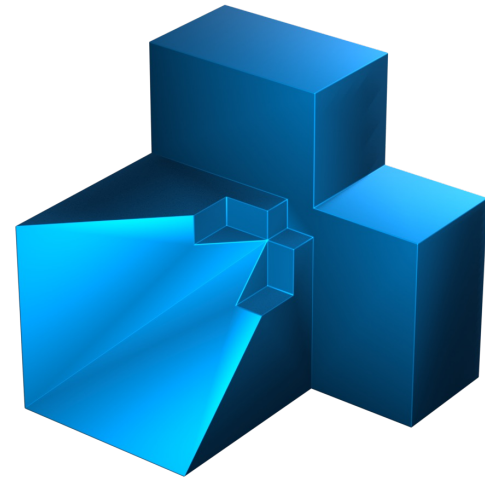
My journey with LLMs

- **June 2025:** these tools are just waste of time, it makes no sense to pay money for ChatGPT...
- **August 2025:** GPT-5 published; the first version that genuinely helps me with my work
- **Autumn 2025:** coding agents are a gamechanger!
 - Cursor → Claude Code → **Codex**

My journey with LLMs

- **Early 2026:** the first major breakthroughs
 - ***GPT-5.2 + Codex*** are solving our open research questions
 - clearly not just “combining known results” but coming up with brand new proof techniques

2-coloring cycles



- **Question:** 1-round randomized distributed algorithms in cycles, 2-coloring, fraction of monochromatic edges?
- **Prior:** 0.20 ... 0.25
- **New:** 0.23879 ... 0.24118
- **Techniques:** *new lower-bound technique*, graphs, symmetries, SDPs, solvers

My journey with LLMs

- **Early 2026:** the first major breakthroughs
- But most of the mathematical claims were garbage — AI produces **slop** way faster than anyone can verify
- **Practical solution:**
 - *ask AI to formalize the proof in Lean 4*
 - only pay attention to the claim once AI manages to convince Lean!

**AI +
formalization**

Formalization & Lean

- **Formalization of mathematics:**
 - *mathematical proof $\approx$ program code*
 - proof verification $\approx$ type checking
- **Could be done in principle, seldom done in practice**
 - typical human proofs $\approx$ pseudocode
 - formalized proofs $\approx$ algorithm implementations in low-level programming languages

Formalization & Lean

- There are many **proof assistants** that help humans with writing formal proofs
- The most widely-used proof assistant today is **Lean 4**

Formalization & Lean

- **Two parallel developments make all this relevant today:**

1. formalization is finally *gaining popularity* in mathematics community: large chunks of undergraduate-level mathematics already formalized in *Lean Mathlib*
2. in *AI-assisted* mathematics easy verification of proofs is especially relevant, and AI agents can also *do formalization for you*

LLMs & Lean

- **Early 2025:** chatbots barely know the syntax of Lean 4

LLMs & Lean

- **Early 2025:** chatbots barely know the syntax of Lean 4
- **Early 2026:** it often takes days (and lots of tokens) but eventually AI agents can produce a formal proof
 - critical: use e.g. Codex coding agent so that the LLM can *interact with the Lean compiler* and get immediate feedback

LLMs & Lean

- **Early 2025:** chatbots barely know the syntax of Lean 4
- **Early 2026:** it often takes days (and lots of tokens) but eventually AI agents can produce a formal proof
- **Today:** automatic “vibe-formalization” is essentially solved

Caveats

- **These are fundamentally different use cases not to be confused:**
 1. produce a *high-quality* library of *reusable* mathematical definitions and proofs
 2. ask chatbots to do a quick *vibe-formalization* as a sanity-check so you know whether the AI slop you got is worth human attention

Caveats

- Lean can check the proof for you
- Lean **cannot** check that the formal version of the theorem statement is **correctly capturing what you wanted to prove!**
- There are also all kinds of loopholes that a proof might abuse...

LLMs & Lean

- **In spite of all caveats, this kind of workflow has been very effective:**
 - ask Codex + GPT to solve a problem
 - once they claim a proof,
ask Codex + GPT to formalize it in Lean 4
("in Palomar style, running all Palomar checks")
 - only pay attention to the proof once Lean is happy

Case study: quantum symmetry breaking

Case study

- **Classical LOCAL:** *there is no $O(1)$ -round algorithm for 3-coloring cycles*
 - fastest possible algorithm: $O(\log^* n)$ rounds
- **Quantum-LOCAL:** does quantum computation and communication help?
- This has been open since 2009 when the quantum-LOCAL model was first introduced...

Case study

- **Conjecture:** there is no quantum advantage
- Some promising progress towards proving a lower bound, but not quite there...
- On September 8, Tom Gur and Longcheng Li posted a new lower bound on arXiv, so the **conjecture seemed to be even stronger...**

Case study

- **September 8:** paper appeared on arXiv
- **Same day:** we asked Codex + GPT-6 to prove a stronger *lower bound*
- **In a few hours:** GPT-6 gave an *upper bound*, disproving the conjecture!
- **In a few hours:** everything formalized in Lean

Case study

- **Some weeks:** humans working towards understanding the proof and writing it up properly — ***"by humans, for humans"***
- We kept sanity-checking our writing by asking AI tools to **formalize also our versions of the proofs** in Lean 4
 - tens of pages of complicated math formalized routinely in a matter of hours...

Case study

arxiv.org/abs/
2609.26788

Quantum Advantage for Distributed Symmetry Breaking

Maxime Flin · Aalto University, Finland

Longcheng Li · University of Cambridge, UK

Jukka Suomela · Aalto University, Finland

Abstract. We present a distributed quantum algorithm that 3-colors cycles in $O(1)$ rounds, with high probability. It follows that all locally checkable labeling problems (LCLs) that have round complexity $O(\log^* n)$ in the classical LOCAL model can be solved in $O(1)$ rounds in the quantum-LOCAL model, with high probability; this includes problems such as maximal independent set and maximal matching in bounded-degree graphs. This presents the first *natural* examples of graph problems with an asymptotic distributed quantum advantage for the LOCAL model; all prior examples that separate LOCAL and quantum-LOCAL are artificial problems constructed merely for the sake of demonstrating quantum advantage.

neighbors. Then we do the same for nodes of color 5, and nodes of color 4. After this, the node has decided its final color and can stop.

4. Nodes that did not make it to the finalization phase remain undecided and start again from step 1.

7 AI Methodology

Our main result was discovered using OpenAI GPT-6 in Codex. Our starting point was the *lower-bound* result in [GL26], and we tried to use the LLM to prove a *stronger* lower-bound result; the first step would be to extend the lower bound from 4 colors to an arbitrarily large number of colors. To our surprise, GPT-6 showed that this is *not* possible: there is, actually, an algorithm.

To the best of our knowledge, [GL26] was the critical human-supplied ingredient that enabled the breakthrough for the AI tools. We have tried to study the same question extensively (also with the latest models such as OpenAI GPT-6 and Claude Fable 5.1), but there was no tangible progress until [GL26] appeared.

Once [GL26] was available, the task was apparently easy for the LLMs. The AI agent proved the equivalents of [Result 1](#) and [Result 2](#) in a few hours, and in regular OpenAI API pricing the cost would have been in the ballpark of 50 euros (much less than that using the monthly ChatGPT subscription).

Beyond the use of AI for initial discovery, we have used AI tools extensively to ensure the correctness of the result—in particular, many versions of [Result 1](#), as well as the corresponding negative result for 4-coloring from [GL26], were formalized in Lean 4 with the help of GPT-6 and Codex. We have also used AI tools to e.g. better understand the proof, explore alternative proof techniques, simplify the proof, explore connections with finitely-dependent colorings. In addition to Codex and GPT-6, we have also used Claude Code and Fable 5.1.

While we used AI tools extensively to discover the proof, **this paper is entirely written by humans, for humans.** In writing, AI was only used to correct spelling errors and grammar mistakes and to verify the proofs that we wrote. To ensure a clean separation between AI slop and human-authored material, we had two Git repositories, one where the AI agents worked, and one where this manuscript was written.

arxiv.org/abs/
2609.26788

Reflections

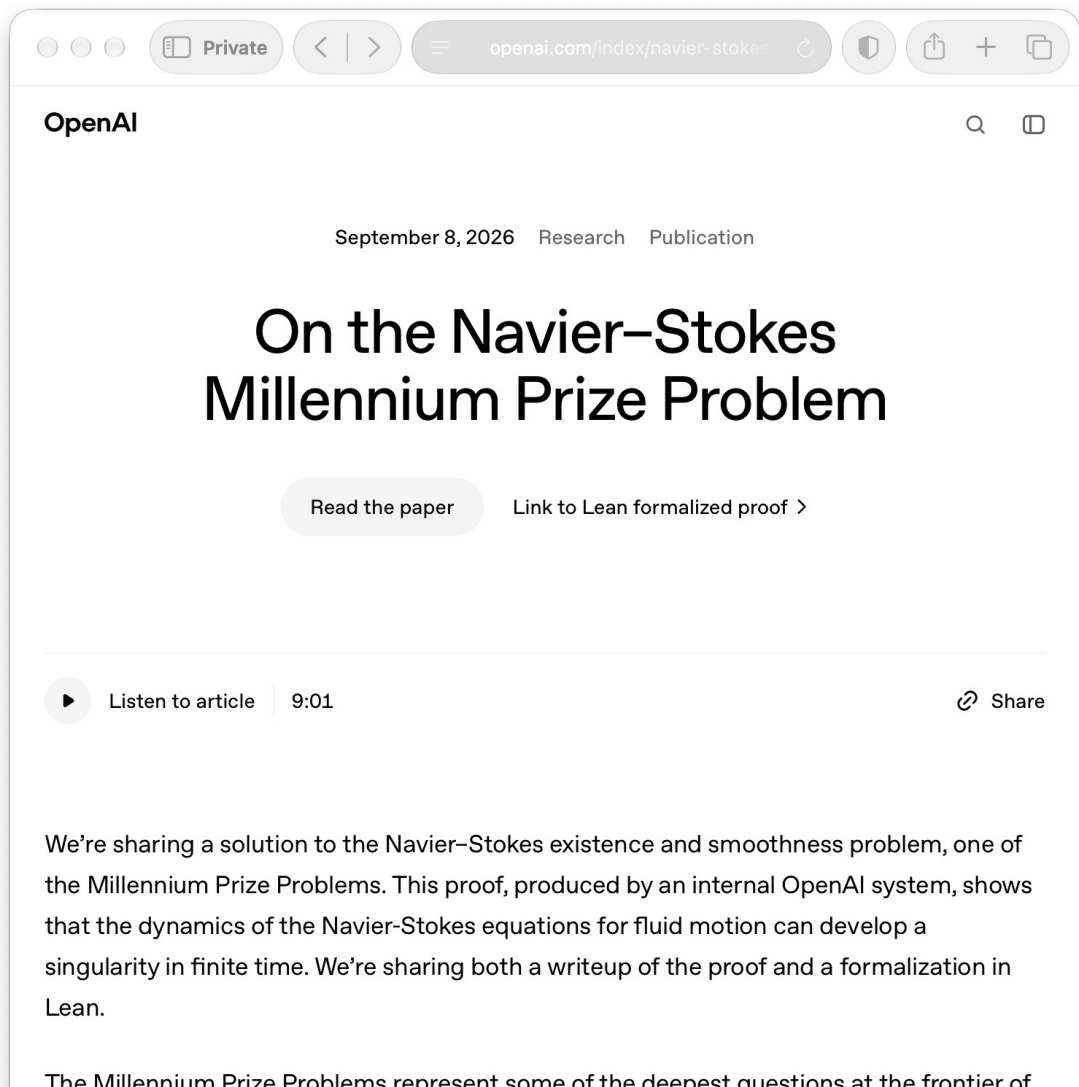
Right research area

- In early 2026, this seemed to be the right area of TCS for this kind of work
- **TCS in general:** key definitions and key results missing from Lean Mathlib
- **Distributed algorithms:** same issue, but it is not so critical!
 - *“an algorithm is a function such that ...”*

Right research area

- This gave us a **head start** with learning these tools
- But LLMs are now so good that this distinction seems to **no longer matter**

Problem-solving solved?



OpenAI

September 8, 2026 Research Publication

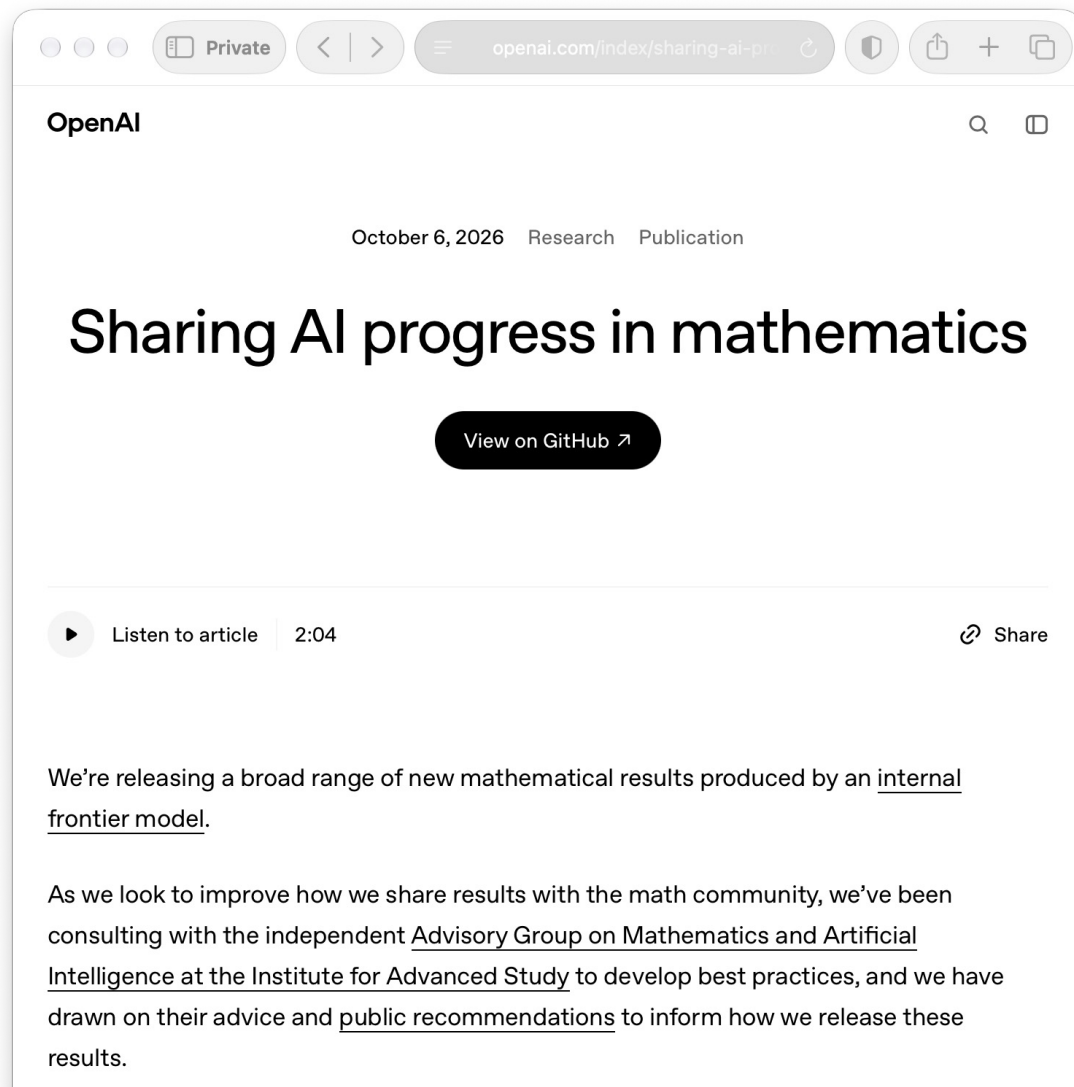
On the Navier–Stokes Millennium Prize Problem

[Read the paper](#) [Link to Lean formalized proof >](#)

▶ Listen to article 9:01 [Share](#)

We're sharing a solution to the Navier–Stokes existence and smoothness problem, one of the Millennium Prize Problems. This proof, produced by an internal OpenAI system, shows that the dynamics of the Navier-Stokes equations for fluid motion can develop a singularity in finite time. We're sharing both a writeup of the proof and a formalization in Lean.

The Millennium Prize Problems represent some of the deepest questions at the frontier of



OpenAI

October 6, 2026 Research Publication

Sharing AI progress in mathematics

[View on GitHub ↗](#)

▶ Listen to article 2:04 [Share](#)

We're releasing a broad range of new mathematical results produced by an [internal frontier model](#).

As we look to improve how we share results with the math community, we've been consulting with the independent [Advisory Group on Mathematics and Artificial Intelligence](#) at the Institute for Advanced Study to develop best practices, and we have drawn on their advice and [public recommendations](#) to inform how we release these results.

Next level?

- It is time to take steps **beyond mere problem-solving**
- If computers can solve our problems ... what are **even bigger things** we could tackle in human–computer collaboration?

Next level?

- 1. Theorems as raw data**
- 2. Theory-building**
- 3. Robust foundations**

Theorems as raw data

- Earlier, every non-trivial theorem was precious
 - so much effort that it better lead to a paper...
- What if we can mass-produce thousands of theorems and treat them as **raw data**?
 - not interesting individually
- What is interesting is phenomena that emerge when we **study this data as a whole**

Theory-building

- Proving a theorem is, in the end, a well-defined task, and an AI agent “knows when it is done”
- **Theory-building is ill-defined and open-ended**
- This is an area where human–computer collaboration could excel

Robust foundations

- We can now afford to **re-prove decades worth of prior results**, against all possible variants of our model definitions
- We can have all of that also **formalized**
- We can put all that in **databases**, build user interfaces, create interactive visualizations etc. for exploring all of that

Next level?

- 1. Theorems as raw data**
- 2. Theory-building**
- 3. Robust foundations**

Room for human–computer collaboration

Opportunities for combining AI slop with rigorous formalized mathematics

Challenges

- How do we **reorganize our scientific community** in this new reality?
- How do publications and conferences work?
- How do we manage all that **backlog** of chatbot-generated results that no human has yet had time to digest?

Challenges

- What happens in your own research community the day **OpenAI releases its next-generation model...**
- ... and dozens of students, postdocs, and professors all around the world feed **the same standard open questions** to it, get the same solutions, and put lots of time and effort in writing up the same solutions?

Summary

- **Three timely intertwined themes:**
 1. Need for robust foundations of distributed computing theory
 2. Modern AI tools
 3. Automatic formalization of mathematics
- Could we let computers do most of our problem-solving so that we can ***focus on bigger things?***

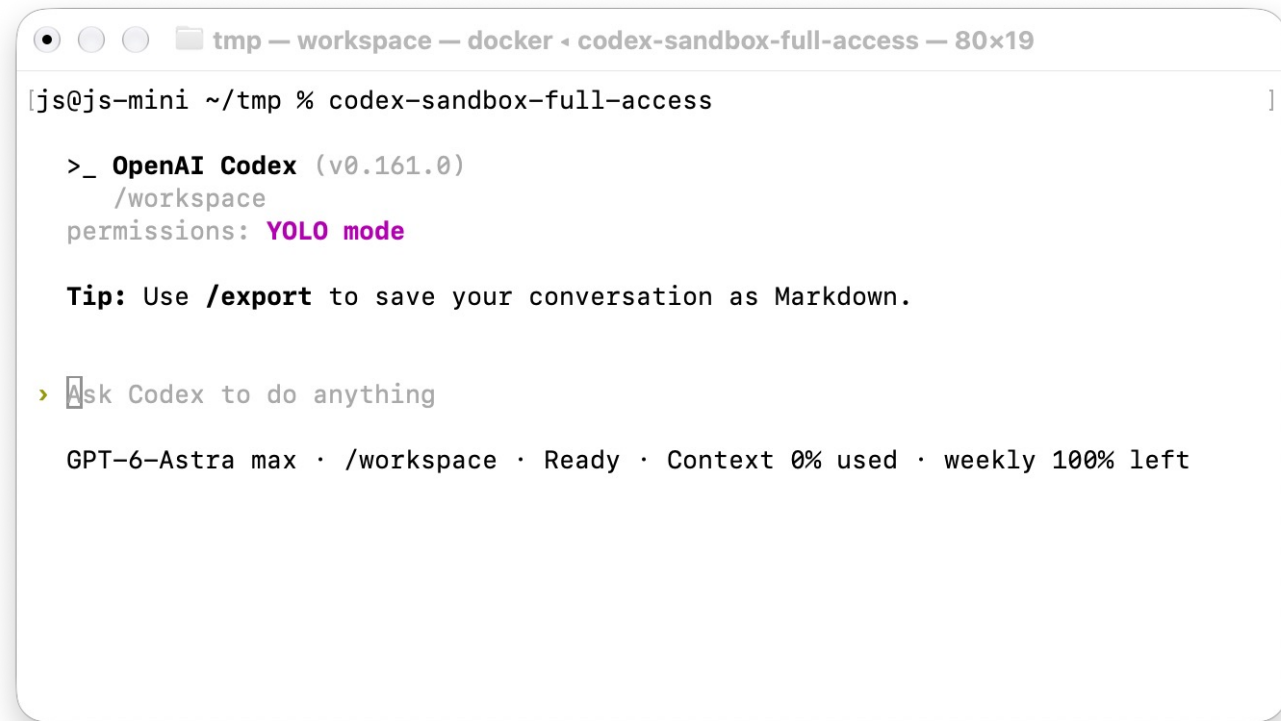
Bonus slides:

My setup

My tools

- ChatGPT Pro
(229 €/month)
- Codex + GPT-6
- Docker sandbox:

github.com/suomela/cursor-sandbox



The screenshot shows a terminal window titled "tmp — workspace — docker • codex-sandbox-full-access — 80x19". The prompt is "[js@js-mini ~/tmp % codex-sandbox-full-access]". The output shows the OpenAI Codex (v0.161.0) interface with the workspace path "/workspace" and permissions set to "YOLO mode". A tip suggests using "/export" to save the conversation as Markdown. Below this, there is a prompt "Ask Codex to do anything" and a status bar at the bottom indicating "GPT-6-Astra max • /workspace • Ready • Context 0% used • weekly 100% left".

```
[js@js-mini ~/tmp % codex-sandbox-full-access]

>_ OpenAI Codex (v0.161.0)
   /workspace
permissions: YOLO mode

Tip: Use /export to save your conversation as Markdown.

> Ask Codex to do anything

GPT-6-Astra max • /workspace • Ready • Context 0% used • weekly 100% left
```

Best practices

- Just prompt in **Codex**, don't worry too much
- Let it run for hours
- Suggest **multiagent setup**, adversarial audit, "fresh pair of eyes" (just prompt)
- Ask it to formalize in **Lean 4** (in Palomar style)
- Use **Git**

Pitfalls

- Gets stuck in running large *numerical experiments* that will never lead to a proof
- Gets stuck *reformulating* the hard problem as another hard problem
- Solves uninteresting special cases

Pitfalls

- Cheats in Lean by formalizing conditional versions
- Or formalizes the right theorem, but with a different proof

What might help

- Explicitly say what you don't want, ***what counts as a solution***, what doesn't, etc.
- Require that all computational experiments must be high-value and ***explicitly justified***
- Refer to e.g. the prompt used for the *Cycle Double Cover* proof
- Prompt for a prompt