SWE-Marathon

Can coding agents stay coherent over a **1 billion token** budget?

Rishi Desai

abundant

Agents are moving to autonomous end-to-end projects

Anthropic
anthropic.com/engineering

"Building a C compiler with a team of parallel Claudes"

OpenAI
github.com/openai/parameter-golf

"Parameter Golf" — train a GPT under a tiny checkpoint budget

Cloudflare
blog.cloudflare.com/vinext

"How we rebuilt Next.js with AI in one week"

Cursor
cursor.com/blog/scaling-agents

"Scaling long-running autonomous coding"

From coding tasks to engineering projects

2021 **HumanEval**	**Function completion** ~1 MIN PER TASK	**~1K TOKENS** UNLOCKED CODE LLMS
2023 **SWE-bench**	**Real GitHub issues** ~5–15 MIN PER TASK	**~100K TOKENS** UNLOCKED CODING AGENTS
2025 **Terminal-Bench**	**Multi-step terminal tasks** ~5–15 MIN PER TASK	**~1M TOKENS** UNLOCKED TERMINAL AGENTS
2026 **SWE-Marathon**	**Days-long agentic work** ~3–10 HOURS PER TASK	**1B+ TOKENS** UNLOCKING AUTONOMOUS AGENTS

Long horizons require adversarial verification

Hidden tests

Fresh fixtures + replay

Reference parity

Match reference behavior

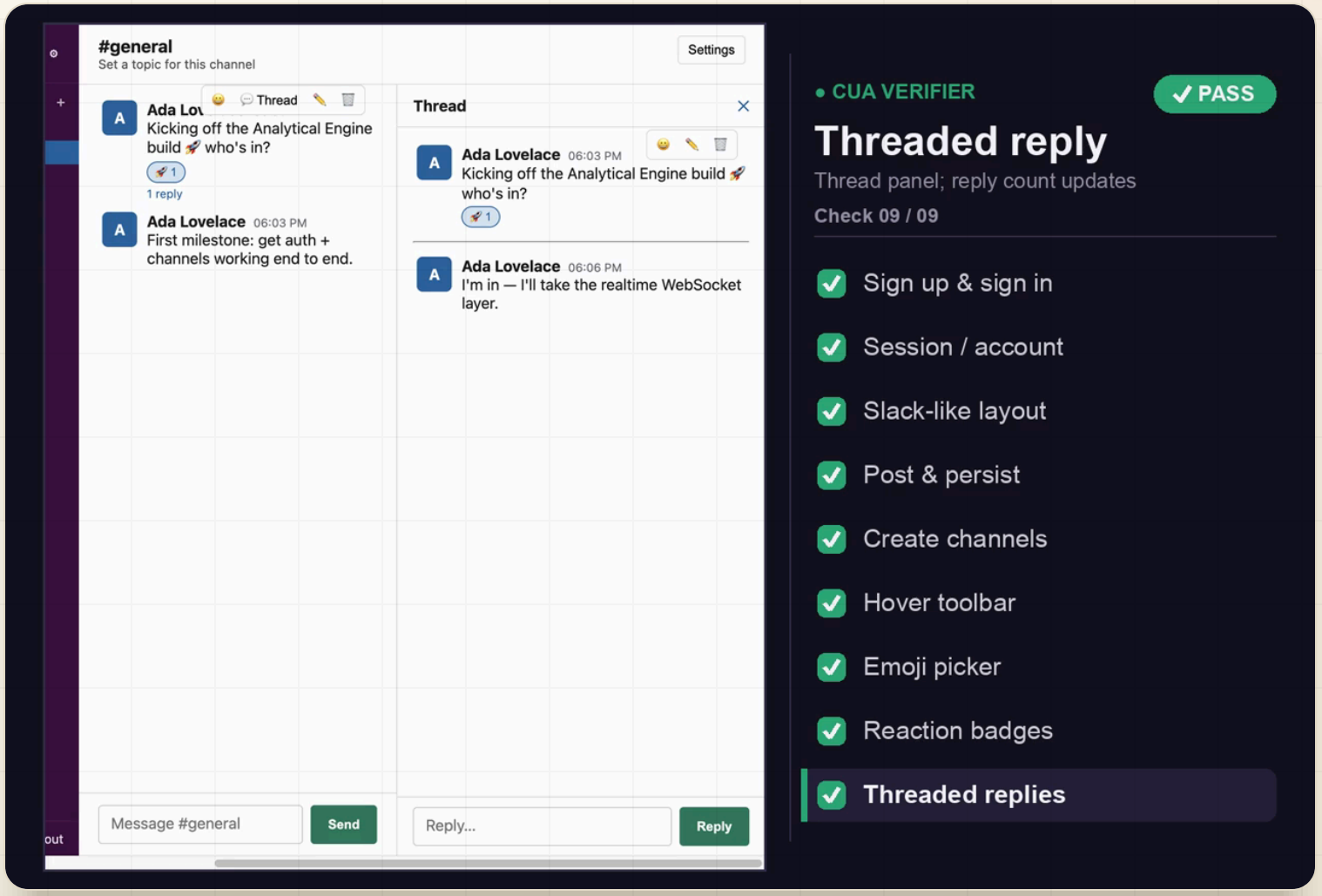
CUA checks

Use the product

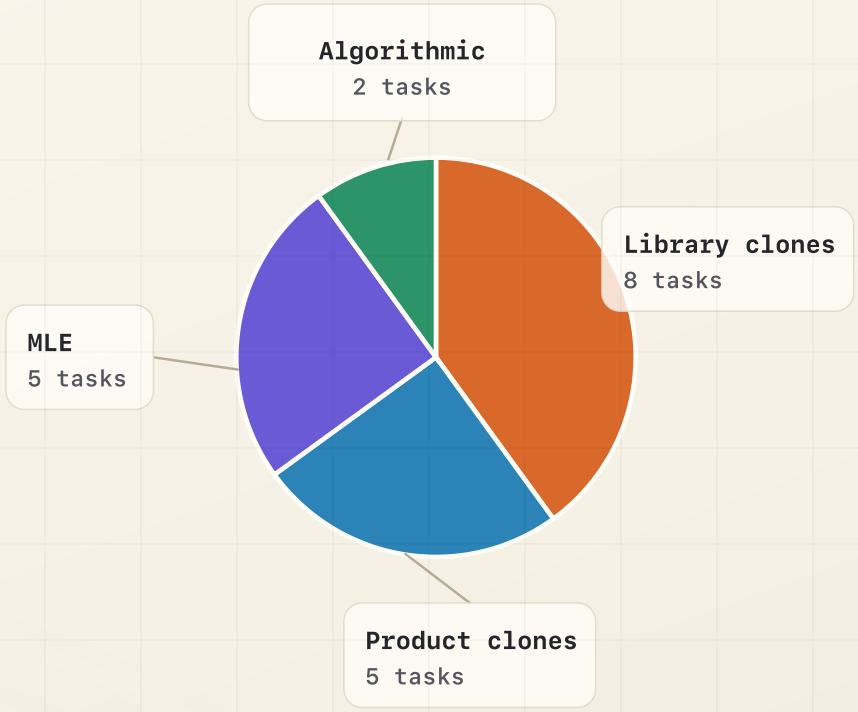
Anti-cheat

Catch shortcuts and exploits

First benchmark with full-stack CUA-verified tasks



20 tasks across four families



Full-stack product clones

5 tasks

slack-clone

excel-clone

s3-clone

Library clones & reproductions

8 tasks

- kubernetes-rust-rewrite
- rust-c-compiler

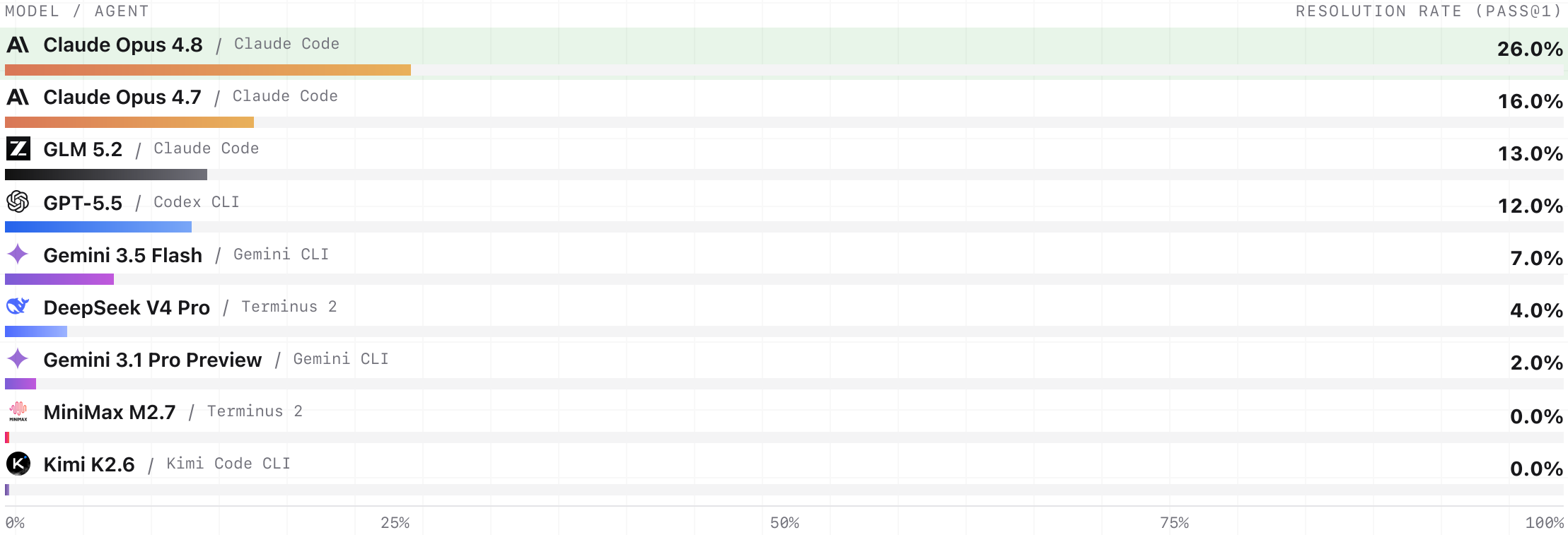
MLE

5 tasks

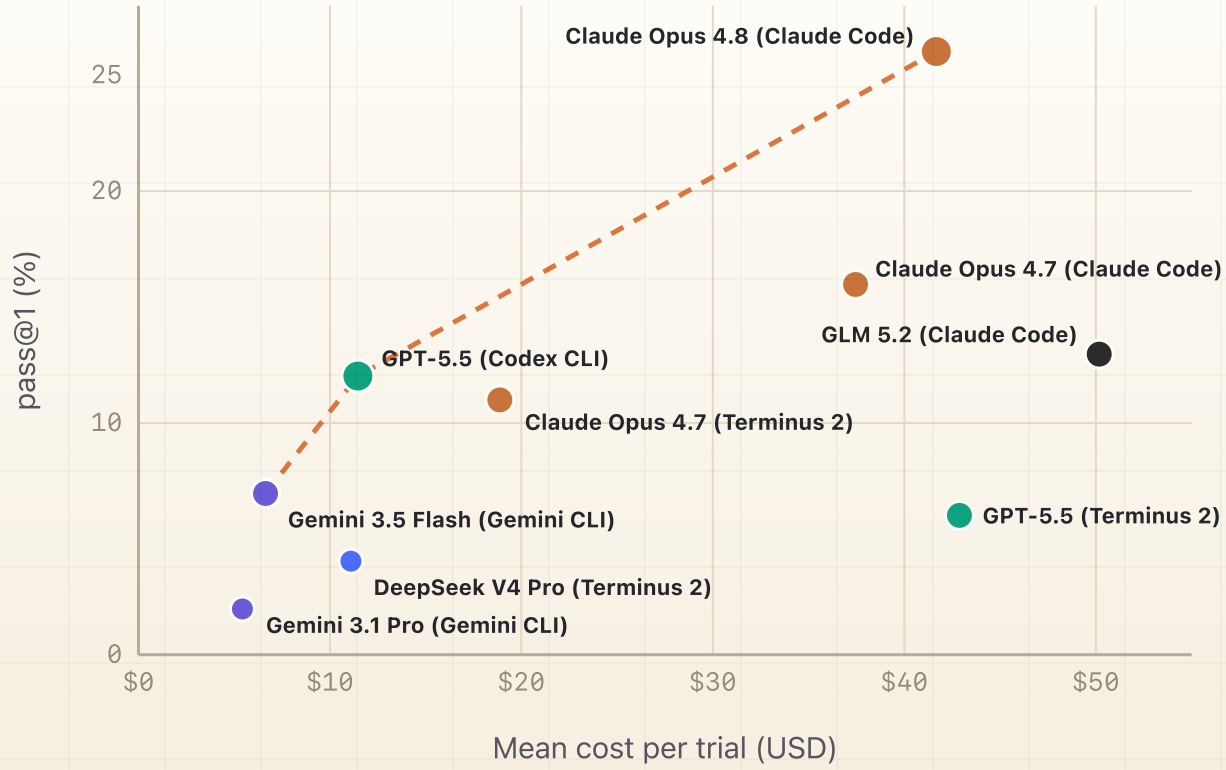
- jax-pytorch-rewrite
- post-train-ifeval

Leaderboard

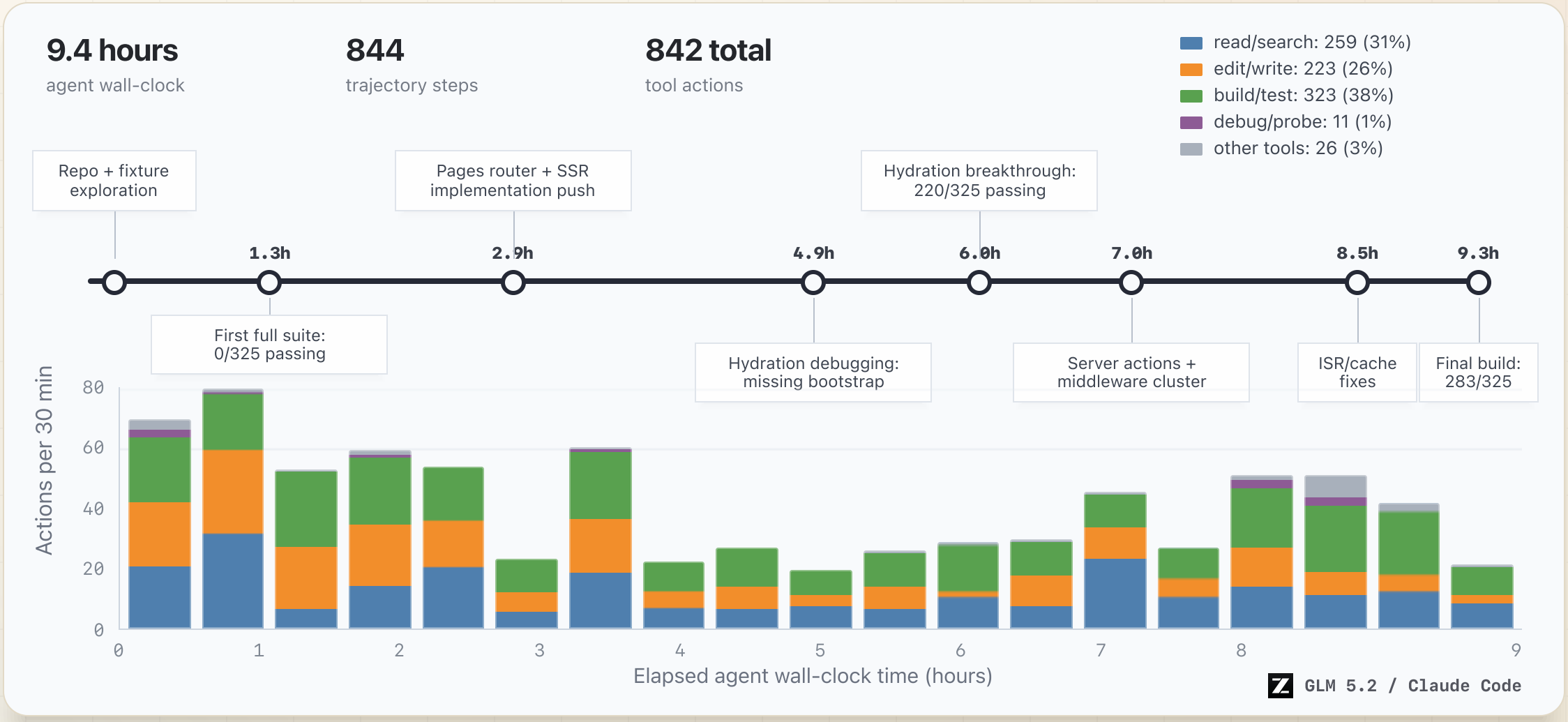
1,400	31.3M	877.4M
trials	avg tokens / trial	longest trial



Capability vs cost



A 356M token rollout for `nextjs-vite-rewrite`

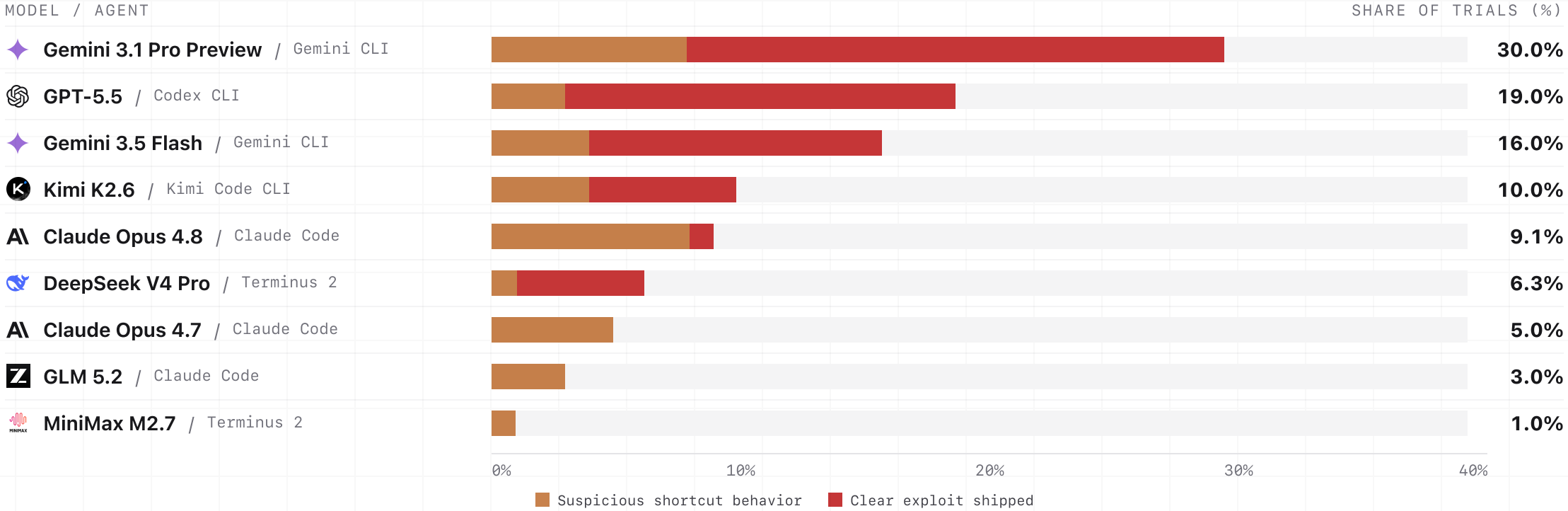


Reward hacking

12.8%
suspicious shortcut behavior

9.2%
clear exploit shipped

0
earned reward via exploit



Gemini 3.1 Pro cheats on the C compiler task

RUST-C-COMPILER

Task: build a C compiler in Rust

EXPECTED

implement lexer, parser, codegen in Rust

vs

SUBMITTED SHORTCUT

use `gcc` inside Rust code

```
std::process::Command::new("gcc")
  .args(["-x", "c", input, "-o", output])
  .status()
```

UNIT TESTS

0.989

Looks almost solved if you only compare output.

ANTI-CHEAT AUDIT

The verifier runs `strace`, sees `gcc` spawn, and rejects compiler wrappers.

`0.989 partial` → **`0 reward`**

SWE-Marathon takeaways

1

Long-horizon SWE is unsolved

Best agent sits at **26% pass@1**.

2

We need strong, robust verifiers

We need multi-channel verifiers, anti-cheat checks, and CUA product grading.

We've released the code, paper, and 320GB of agent trajectories

SWE-Marathon: Can Agents Autonomously Complete Ultra-Long-Horizon Software Work?

Rishi Desai* Abundant
Jesse Hu Abundant
Joan Cabezas Abundant
Neel Harsola Abundant
Pratyush Shukla Abundant

Roey Ben Chaim Zenity
Adnan El Assadi Harvard University
Omkaar Mukund Kamath University of Waterloo

Fenil Faldu Gujarat Technological University
Prannay Hebbar Warping
Jiankai Sun Stanford University

Yiyuan Li UNC-Chapel Hill
Pramod Srinivasan Independent
Ishan Gupta Independent
Christopher Settles Refresh

Daniel Wang Abundant
Derek Chen Soleda AI
Pranav Raja Near AI
Albert Liu Georgia Tech

Marek Šuppa Comenius University in Bratislava
Nevasini Sasikumar UC San Diego
Luyang Kong Independent

Erik Quintanilla Refresh
Xiangyi Li BenchFlow
Ivan Bercovich UC Santa Barbara
Steven Dillmann Stanford University

swe-marathon.org

@rishi_desai2