Correct Is Not I/O Efficient: Characterizing I/O Behavior of LLM Agents on Scientific Data

Gnosis Research Center @ ILLINOIS TECH

Izzet Yildirim Jie Ye Jaime Cernuda Xian-He Sun Anthony Kougkas

Contact: iyildirim@hawk.illinoistech.edu

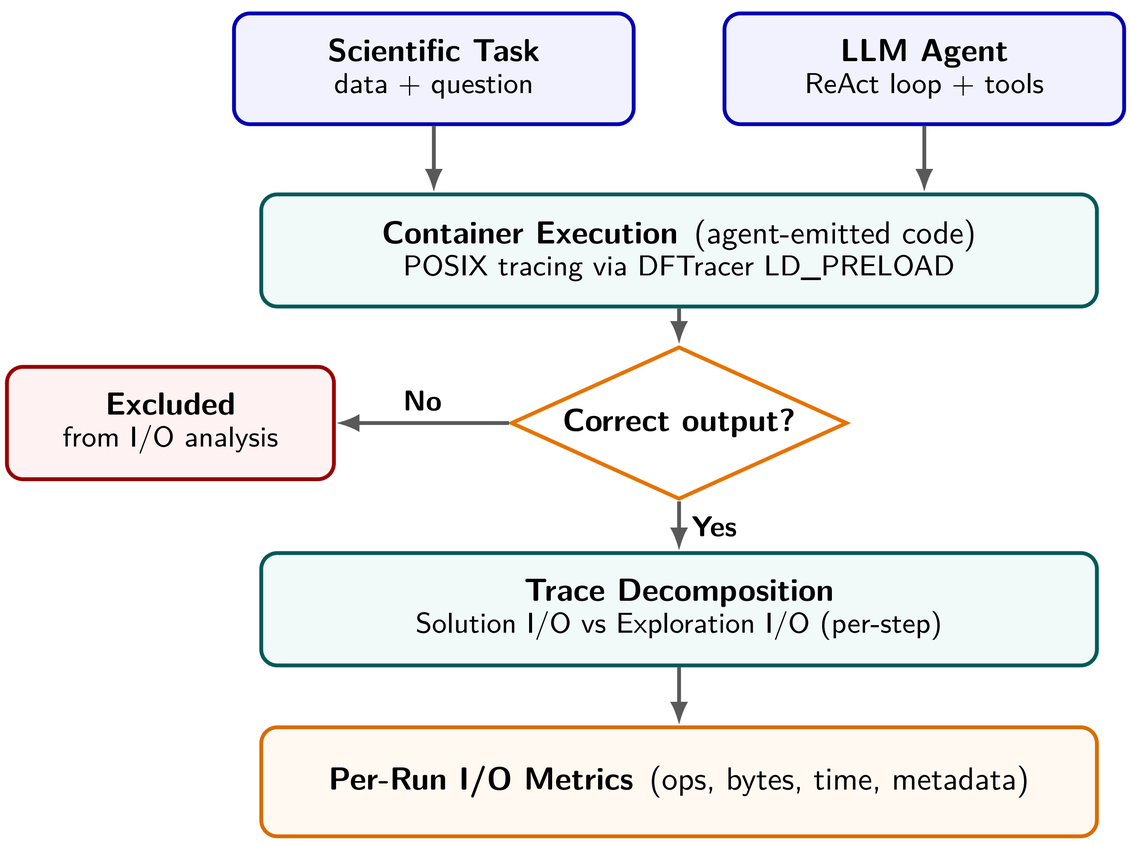
Introduction

LLM coding agents are increasingly used to generate scientific workflows, but evaluations stop at output correctness. For data-intensive science, this is incomplete: two workflows can produce the same valid result while placing different pressure on storage systems. We study I/O behavior in correctness-passing LLM-generated scientific workflows.

Contributions:

- A **benchmark** of scientific data tasks paired with expert implementations and a containerized execution harness.
- **POSIX-level tracing** of agent-emitted code via DFTracer LD_PRELOAD inside isolated subprocesses.
- **Per-step decomposition** of I/O into the final solution path versus exploration overhead (failed attempts, inspection, verification, retries).

Architecture



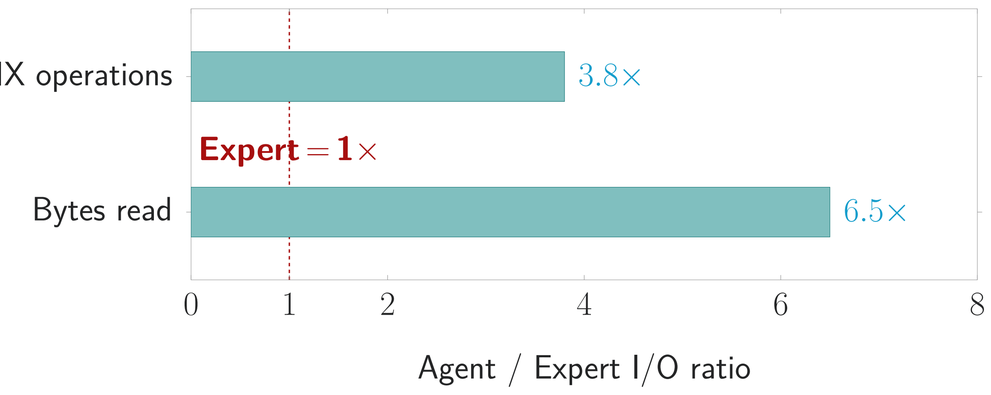
- **Correctness gating** excludes failed runs: "fast because it skipped computation" is not efficient I/O.
- **Trace decomposition** separates solution I/O from exploration I/O (failed attempts, inspection, retries).

Methodology

- **Models:** Claude Sonnet 4.5 (Dense), Gemini 2.5 Flash (Dense), Gemma 4 26B A4B (MoE).
- **Tasks:** 5 tasks spanning climate, neuroscience, and genomics; dataset sizes up to 18 GB.
- **Harness:** Podman container with the scientific Python stack + DFTracer [1]; multi-turn loop over four tools (`list_dir`, `read_file`, `write_file`, `run_shell_command`).
- **Validator:** structural checks + correlation against the expert reference, gating correctness before I/O analysis.
- **Hardware:** IIT **Ares** (32 compute nodes, Intel Xeon Silver 4114, 40 Gbps RoCE) and NCSA **Delta** (8-way A100 GPU nodes, 2 TB RAM) for Gemma 4 inference.

Result 1: Agents Use More I/O Than Experts

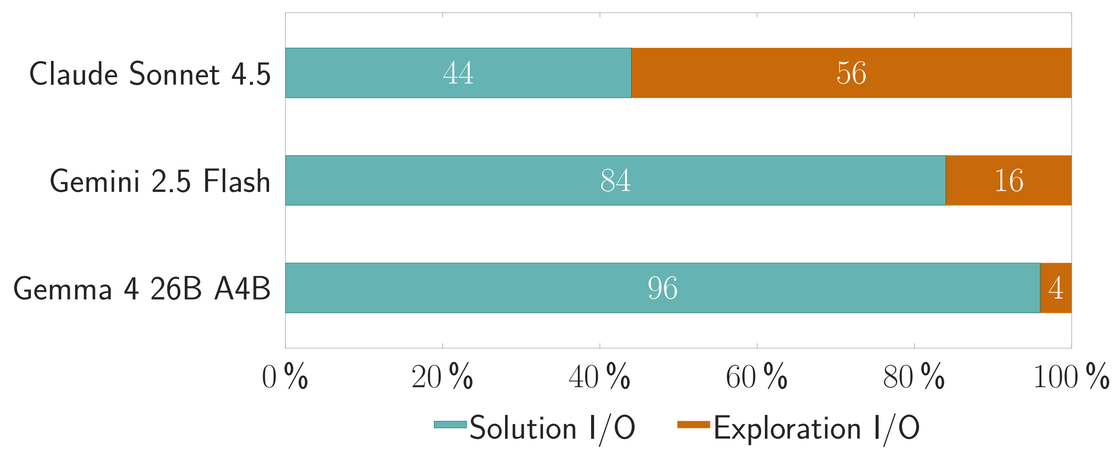
Across the five tasks, **46 of 105 (44%)** agent-generated workflows passed validation. Every passing cell shows the expert implementation using fewer POSIX operations and reading less data than the agent.



- **How to read:** bars show the median agent / expert ratio; the dashed red line marks the expert baseline.
- **Takeaway:** two correct workflows for the same task can place dramatically different pressure on storage; the gap is *I/O strategy* (format, library, materialization), not verbosity.

Result 2: Exploration I/O Varies By Model

For each passing run, I/O is decomposed into *solution* steps (the successful execution path that produced the output) and *exploration* steps (failed attempts, inspection, retries).



- **How to read:** each bar splits a model's bytes into solution-path I/O (teal) and exploration-path I/O (orange).
- **Takeaway:** exploration overhead is real and *varies by model* — correctness-passing runs can diverge sharply once exploration is counted.

Conclusion

Correctness $\neq$ I/O efficiency. Valid agent-generated scientific code issues a median $3.8\times$ more POSIX operations and reads $6.5\times$ more data than expert baselines, with exploration overhead that varies by model. **Storage behavior should be measured directly before deploying agent-generated workflows on shared HPC or cloud storage.**

Acknowledgments & References

This material is based upon work supported in part by the National Science Foundation (NSF) under Award OAC 2411318. This research used the Delta advanced computing and data resource, which is supported by the National Science Foundation (Award OAC 2005572) and the State of Illinois.

[1] Hariharan Devarajan, Loïc Pottier, Kaushik Velusamy, Huihuo Zheng, Izzet Yildirim, Olga Kogiou, Weikuan Yu, Anthony Kougkas, Xian-He Sun, Jae Seung Yeom, and Kathryn Mohror. DFTracer: An analysis-friendly data flow tracer for ai-driven workflows. In *SC24: International Conference for High Performance Computing, Networking, Storage and Analysis*. IEEE, 2024.