Enhancing AI-Assisted Debugging in Parallel Programs via Trace-Level Provenance

Xulu Chu[1], Yuta Nakamura[1], Tanu Malik[1 2]

[1] School of Computing, DePaul University, Chicago, IL, USA

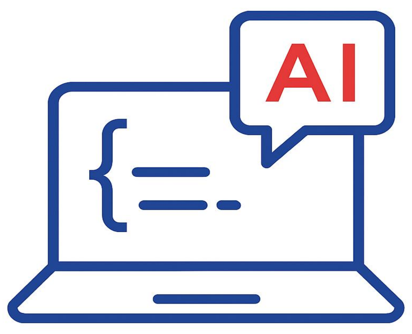
[2] Dept. of Electrical Engineering & Computer Science, University of Missouri, Columbia, MO, USA

INTRODUCTION

AI for Coding Can't Do Everything

Copilot[1] and Cursor[2] excel at code completion and reasoning.

However, they often struggle to explain subtle differences between runs of parallel programs.



The Challenges

- Traditional tools and LLMs often miss key differences in parallel executions.
- LLMs often hallucinate or overlook key runtime differences.

This leads us to ask: **Does providing trace-level differences help LLMs explain why parallel program runs behave differently?**

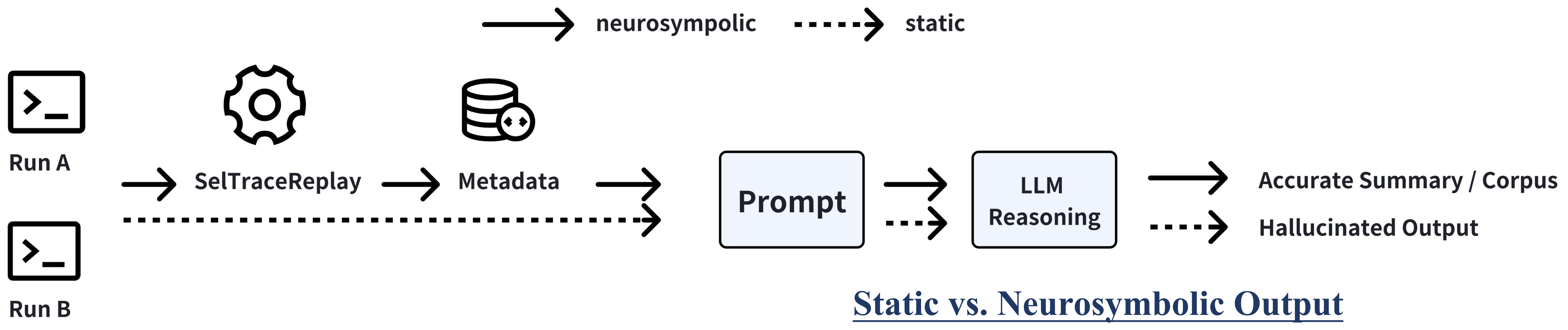
Our Contribution

- **Focus on MPI:** Our method targets the case—MPI programs with nondeterministic behavior.
- **Neurosymbolic + Chain-of-Thought[3] Prompting:**

Combines neurosymbolic (symbolic trace analysis and LLM reasoning) with stepwise CoT prompting. Improves localization accuracy by 30% (Jaccard Similarity).

- **Corpus Collection:** The pipeline accumulates trace-diff/explanation pairs for future LLM fine-tuning.

Example



Source

Code: https://github.com/hypre-space/hypre/blob/master/src/examples/ex7.c

Run A: mpirun -np 4 ./ex7 -n 5 -solver 1 -K **3** -B 0 -C **1** -U0 2 -F 4

Run B: mpirun -np 4 ./ex7 -n 5 -solver 1 -K **4** -B 0 -C **2** -U0 2 -F 4

SelTraceReplay[4]

C:entry:0
hypre_DataExchangeList:neither:32, 40, 43, 44, 48:loop
hypre_PFMGSolve:neither:16, 9:loop
K:entry:0

Metadata

```
{
  "function": "hypre_PFMGSolve",
  "line": 16,
  "divergence_type": "neither",
  "in_loop": false,
  "impact_type": "core",
  "cause": ["-C", "-K"],
  "explanation": "Altered solver logic"
},
```

```
{
  "function": "hypre_DataExchangeList",
  "line": 40,
  "divergence_type": "neither",
  "in_loop": true,
  "impact_type": "side_effect",
  "cause": ["-C"],
  "explanation": "Communication noise"
} ...
```

Prompt

1. Role: You are an expert MPI debugger...

2. Context: Input from Source (and Metadata)

3. Task: Write a concise debugging summary for developers.

For each point, explain:

1. **Core Impact** (eg. logic or control flow)
2. **Side Effect** (eg. communication)

Static vs. Neurosymbolic Output

Static

Input Prompt: Source

Output

1. Divergence by -C

Core Impact: GenerateMatrix() builds a structurally different gradient operator A_C.

Side Effect: Alters the MPI communication pattern during the solve phase.

2. Divergence by -K

Core Impact: GenerateMatrix() generates a numerically different stiffness matrix A_K.

Side Effect: Affects the solver's convergence performance and memory usage.

Issue

- ✗ Misses specific divergences (PFMGSolve, DataExchangeList)
- ✗ Find the wrong divergences (Contains hallucinations)

Neurosympolic

Input Prompt: Source + Metadata

LLM Output

1. Divergence by -C

Core Impact: Alters the smoothing algorithm in hypre_PFMGSolve, switching the core solver path.

Side Effect: Causes corresponding changes in communication patterns within hypre_DataExchangeList.

2. Divergence by -K

Core Impact: Changes the graph coarsening algorithm, affecting the Algebraic Multigrid (AMG) setup.

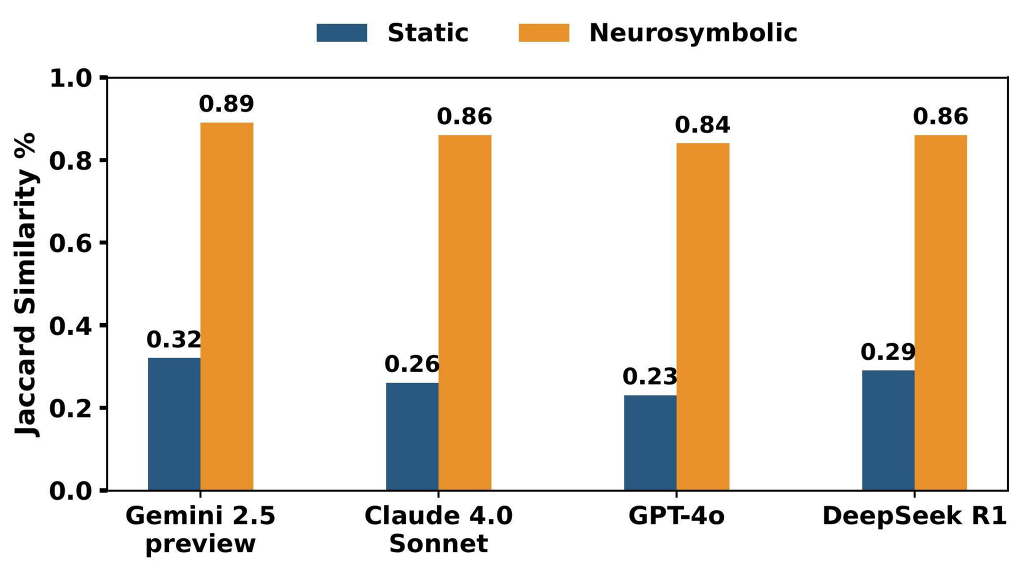
Advantages

- ✓ **Accurate Summary:**
  - Localizes divergences to specific functions and lines
  - Distinguishes input vs communication nondeterminism
- ✓ The output can be **Corpus** to fine tune LLM

EXPERIMENTS

Settings

- **Dataset:** Dataset: 5 MPI modules from the hyper library.
- **LLMs:** Evaluated on leading models: Gemini 2.5 preview, Claude 4.0 Sonnet, GPT-4o, DeepSeek R1.



Results

- Jaccard Similarity: Static: 0.23—0.32 Neurosymbolic: 0.84—0.89
- **30%** improvement in localization accuracy across all models.
- **Fewer hallucinations and false positives** with neurosymbolic prompts.
- GPT—4o vs Gemini: GPT—4o is more likely to hallucinate non-existent trace differences (e.g., hypre_ParCSRRelax:entry:0)

REFERENCES

[1] Cursor AI. 2024. Cursor: The AI-First Code Editor. https://www.cursor.so/.

[2] GitHub. 2021. Introducing GitHub Copilot: Your AI Pair Programmer. https://github.blog/2021-06-29-introducing-github-copilot-ai-pair-programmer/.

[3] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Tallan Pillai, Aitor Lewkowycz Emami, Ed H. Li, and Et Al. 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In Advances in Neural Information Processing Systems. https://arxiv.org/abs/2201.11903

[4] Yuta Nakamura, Xulu Chu, Ignacio Laguna, and Tanu Malik. 2025. Accurate Differential Analysis Using Record and Selective Replay. In Proc. SSDBM '25. ACM. doi:XXXXXXX.XXXXXXX

ACKNOWLEDGEMENTS

This work is supported by the National Science Foundation under grants CNS-1846418 and by the National Aeronautics Space Agency under grant AIST-21-0095-80NSSC22K1485.