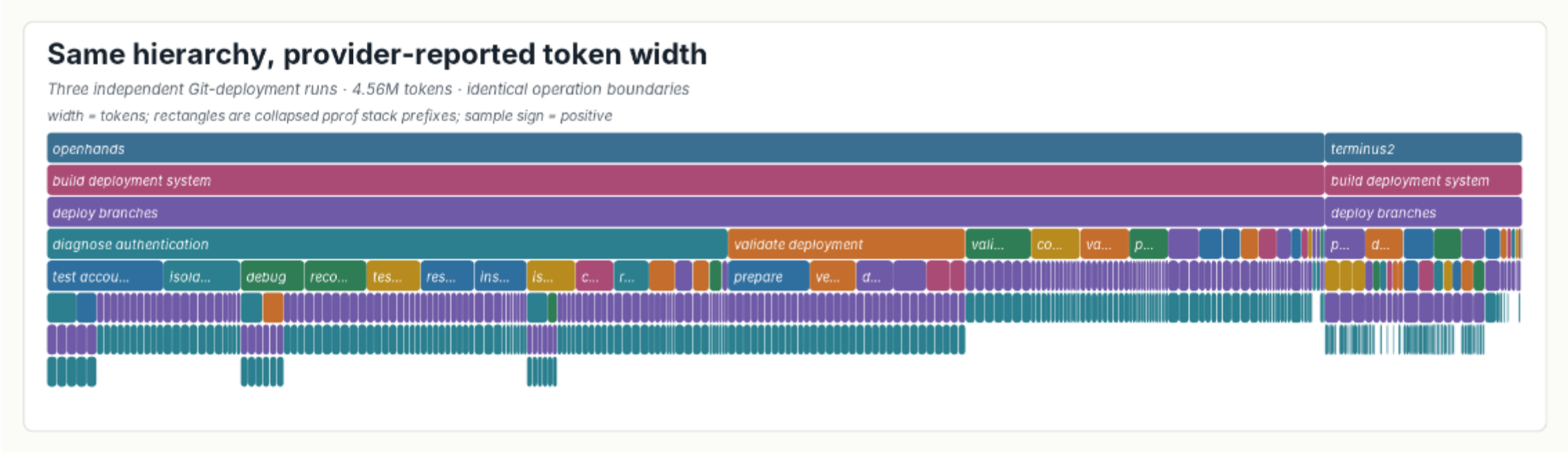
AgentPProf: Semantic Profiler for Long Horizon AI Agents

Attributing agent cost to the work it was doing

Yusheng Zheng et al.

Introduction: Profile responsibilities, not calls



The frame width represents token mass attributed to a responsibility.

Background and Motivation: One responsibility spans three runs

Three independent attempts fail to deliver a password-authenticated endpoint.

`diagnose authentication` accounts for:

Measure	Share
Operations	21%
Elapsed time	37%
Tokens	46%

Counting tool calls alone understates this task’s token cost.

Design: Semantic operation stack model

Record component	Examples
Fields	Project, session, task, action, path, domain
Measures	Tokens, duration, event count
Evidence	Prompt, tool call, recorded system effect

An **operation stack** projects selected fields into a frame sequence.

Identical sequences fold together, summing the selected measure.

Design: Recursive operation segmentation

A trajectory occupies a sequence of steps.

1. Locate transitions where responsibility changes.
2. Name the resulting contiguous intervals.
3. Recurse when an interval contains smaller subtasks.
4. Assign each operation its chain of interval names.

Example: `build deployment system / diagnose authentication`

Implementation: Recorded events to pprof

- A Rust CLI reads Codex and Claude JSONL plus AgentSight recordings.
- Segmentation produces named intervals; validation checks nesting and coverage.
- Deterministic projection sums operation weights into standard pprof frames.

Evidence stays attached: semantic names are inferred; recorded events are retained.

Evaluation · RQ1: Cost attribution across runs

In 41 long-horizon trajectories, semantic stacks fold repeated responsibilities.

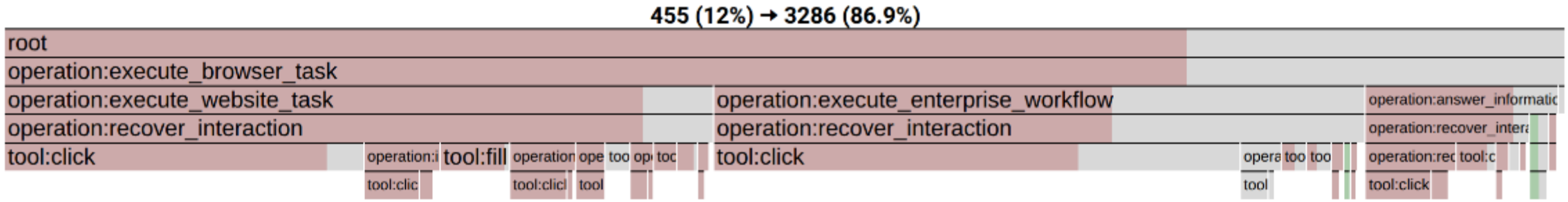
The Git-deployment case locates **diagnose authentication** across three failed attempts:

Operations	Elapsed time	Tokens
21%	**37%**	**46%**

The flame graph exposes cost that raw action counts hide.

Evaluation · RQ2: Recovery explains failed runs

Across **440 web-agent trajectories**, failed runs devote **44.6%** of steps to recovery, versus **12.0%** in successful runs.



Recovery decomposes into repeated searches, verification, navigation, and retries.

Evaluation · RQ3: Human stage agreement

CodeTraceBench: 405 trajectories and 2,948 human-annotated stages.

Method	B^3 F1	Exact boundary F1
Codex segmentation	**0.764**	**0.480**
Statistical recurrence	0.663	0.266
Causal Qwen2.5-3B task stack	0.650	0.257
Raw-action grouping	0.541	—

Labels and scores stay hidden until outputs are fixed.

Evaluation · RQ4: Cost of constructing profiles

For **405 CodeTraceBench trajectories**, segmentation took **37 minutes** with up to four workers.

Mean model use: **29,754 input / 573 output tokens** per trajectory.

Once intervals are fixed, projecting the **27,765-operation** union takes **1.16 seconds** and peaks at **465 MiB**.

Related Work: A complementary profiling axis

Execution traces show **what happened in one run**. Conventional profilers aggregate low-level resource stacks.

AgentPProf adds a **semantic responsibility stack** so tokens, time, and operations can be compared across runs in standard pprof tools.

Conclusion: Agent work becomes a profiling target

Attribute cost to responsibilities, then inspect their evidence.

AgentPPProf supplies stable names, hierarchical attribution, and standard pprof output.

Paper source · Implementation

Questions?