Skills Before Smarts

Harness Engineering for Robust Software-Development Agents

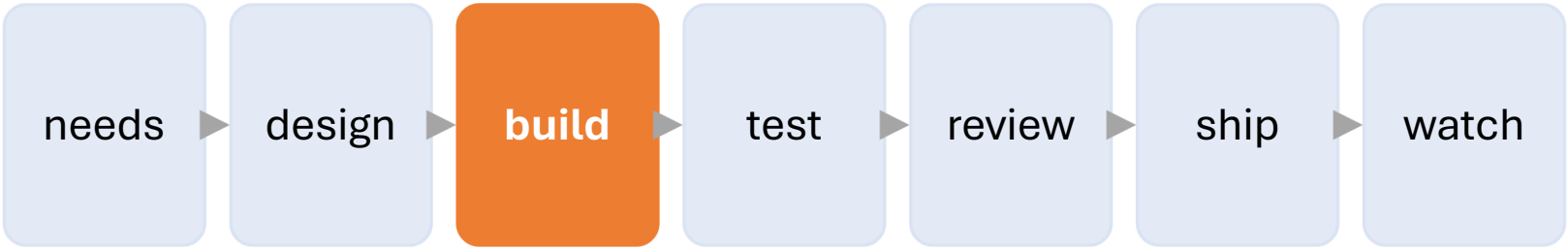
Hariharan Devarajan

Computer Scientist, Center for Applied Scientific Computing (CASC)

MLAC 2026, PACT, Chicago IL

Lawrence Livermore National Laboratory LLNL-CFPRES-2025582 This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344. Lawrence Livermore National Security, LLC (LLNL LDRD 25-ERD-042) NNSA National Nuclear Security Administration

Writing code is one stage of many



LLMs are now in every one of them

The recipe for a hard task is a bigger model

- bigger model
- longer context
- more prompting





the model supplies it all

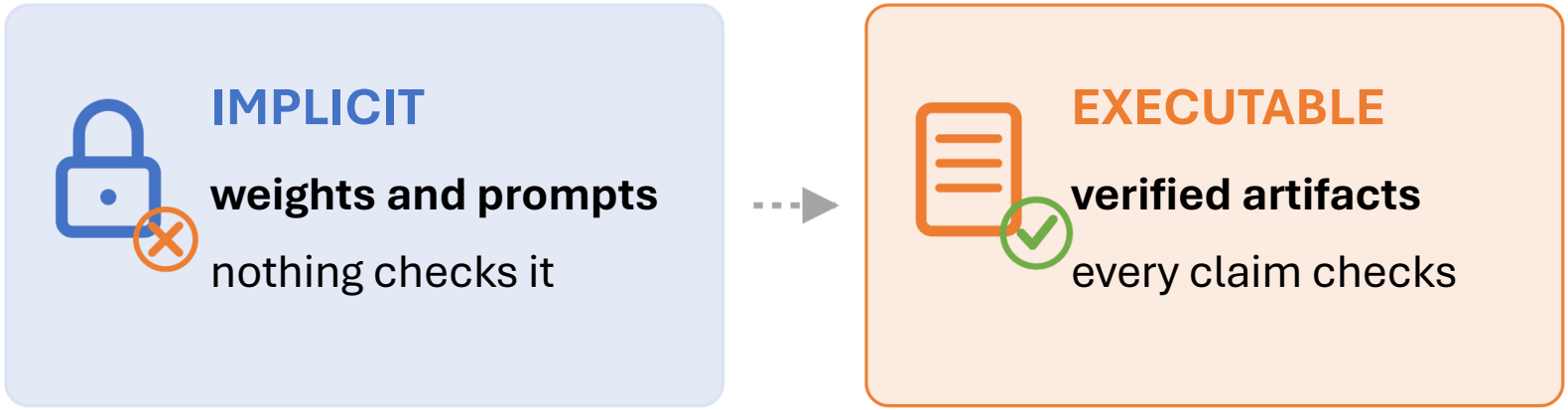
- the algorithm
- the realization
- the conventions
- 'fast enough'

Next: the problem



The gap that limits today's coding agents

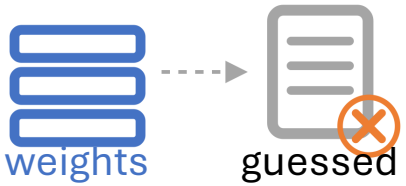
Where reliability is lost



reliability tracks model strength, not engineering

What breaks, concretely

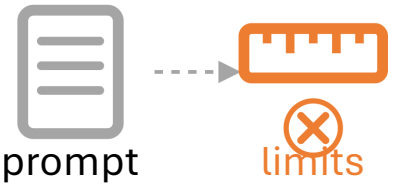
1 RECALLED from weights



no proof it is correct

Recall is not verification

2 NO TARGET scale unsaid



optimizes for plausibility

No target means no aim

3 UNJUDGED no judge



independent axes

Passing is not fast enough

Capabilities first, intelligence second

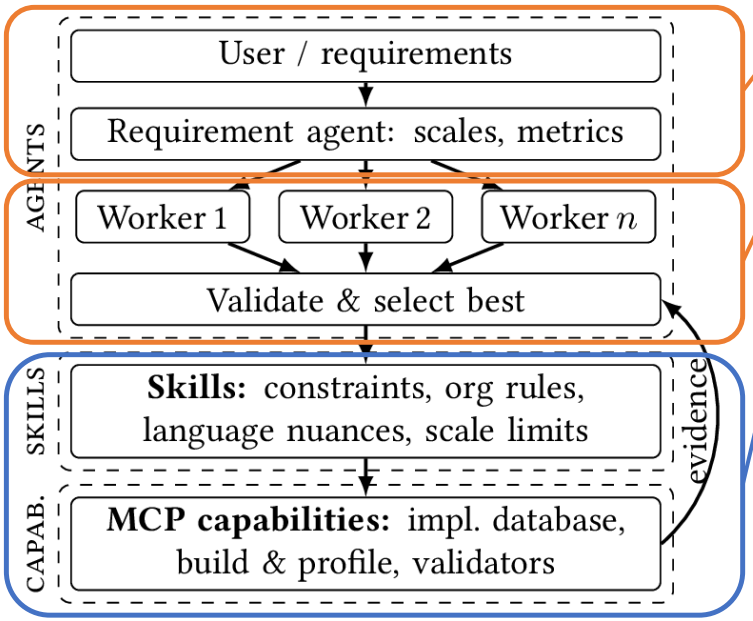


the model composes and selects — it does not recall

Next: the design



Three layers: capabilities, skills, agents



1. Ask first — scales, metrics, tests, constraints
2. Subagents explore; gates and metrics pick the winner
3. Distil into MCP tools + skills: how to do it, what to avoid

fewer iterations each run: less time, fewer tokens

Each layer solves one challenge

1 CAPABILITY
verified



Build algorithms
Profile, and validate



carry evidence

solves C1: recall

2 SKILLS
context bound



constraints
org rules

scale limits



this codebase

solves C2: target

3 AGENTS
search, prove



explore
parallel



only validated

solves C3: judging

Capabilities are verified, not recalled



algorithms

classic problems



build + profile

compile and measure

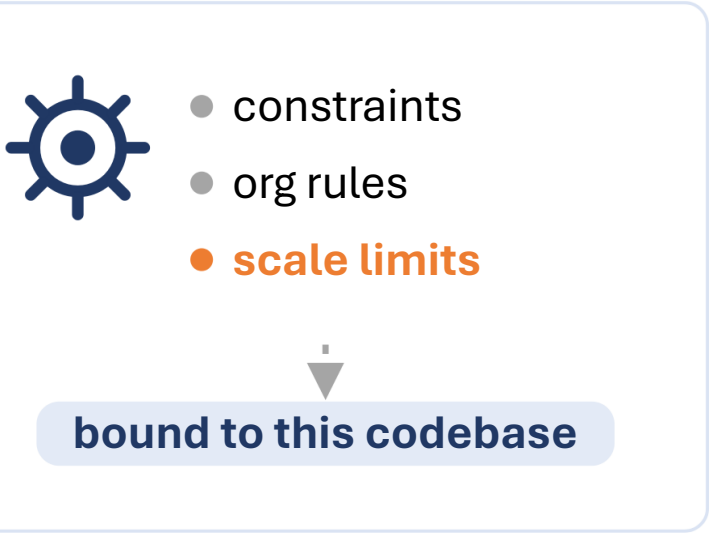


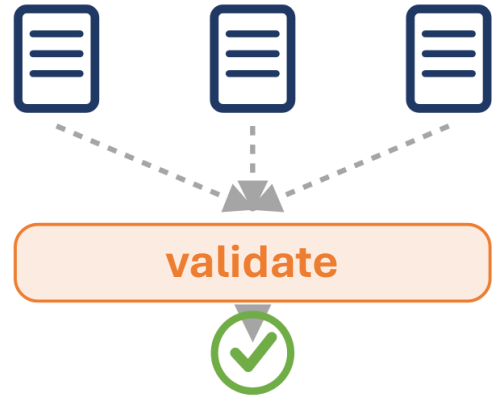
validators

execute under test

every one versioned, every one carrying its evidence

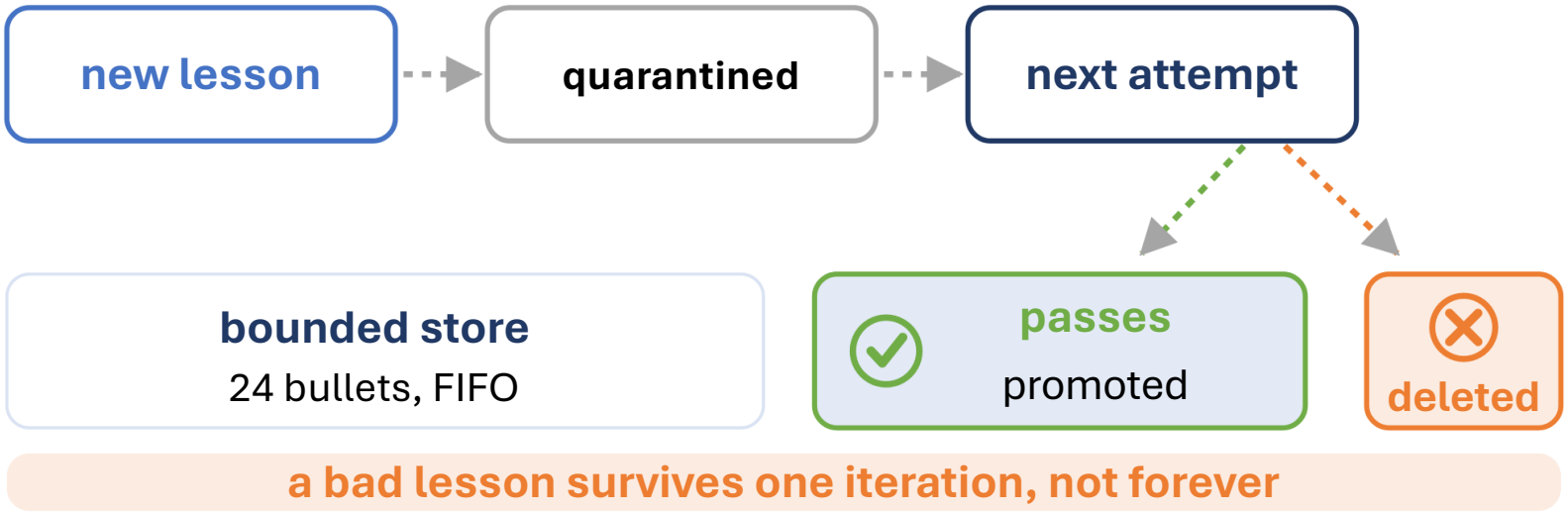
Skills bind context; agents prove





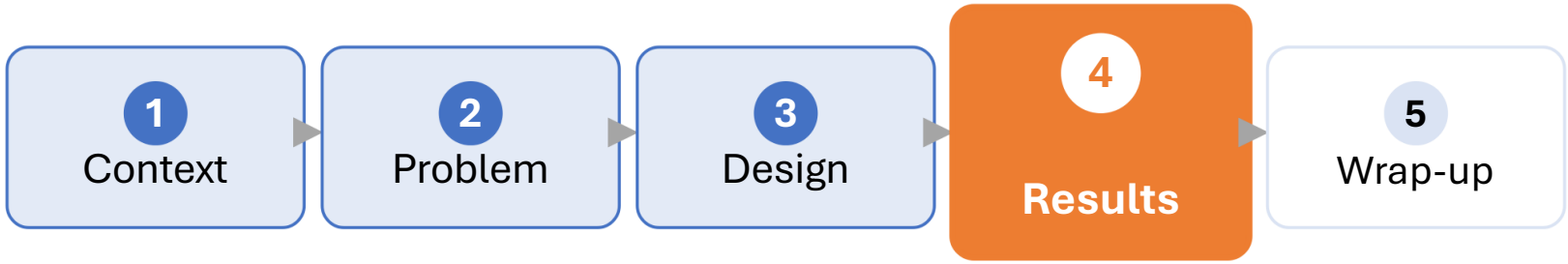
only the proven survive

Lessons must earn their place



a bad lesson survives one iteration, not forever

Next: the evaluation



Same model, same agent – our solution on vs off

The task: build a local filesystem

1 Metadata index



B-tree invariants

2 Object store



layout + placement

3 Session cache



eviction policy

Why it is hard

- invariants must hold
- the three must agree
- the choice depends on load

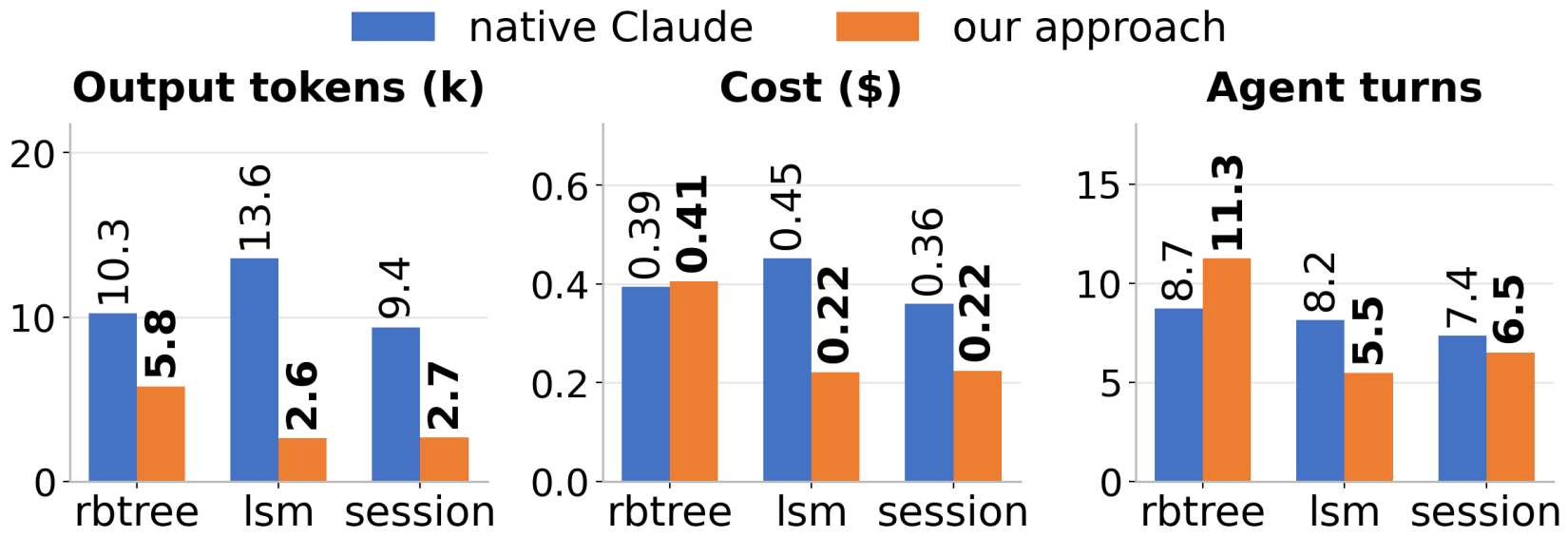
What an LLM reaches for

- recall a textbook version
- make the unit tests pass
- stop - and call it done



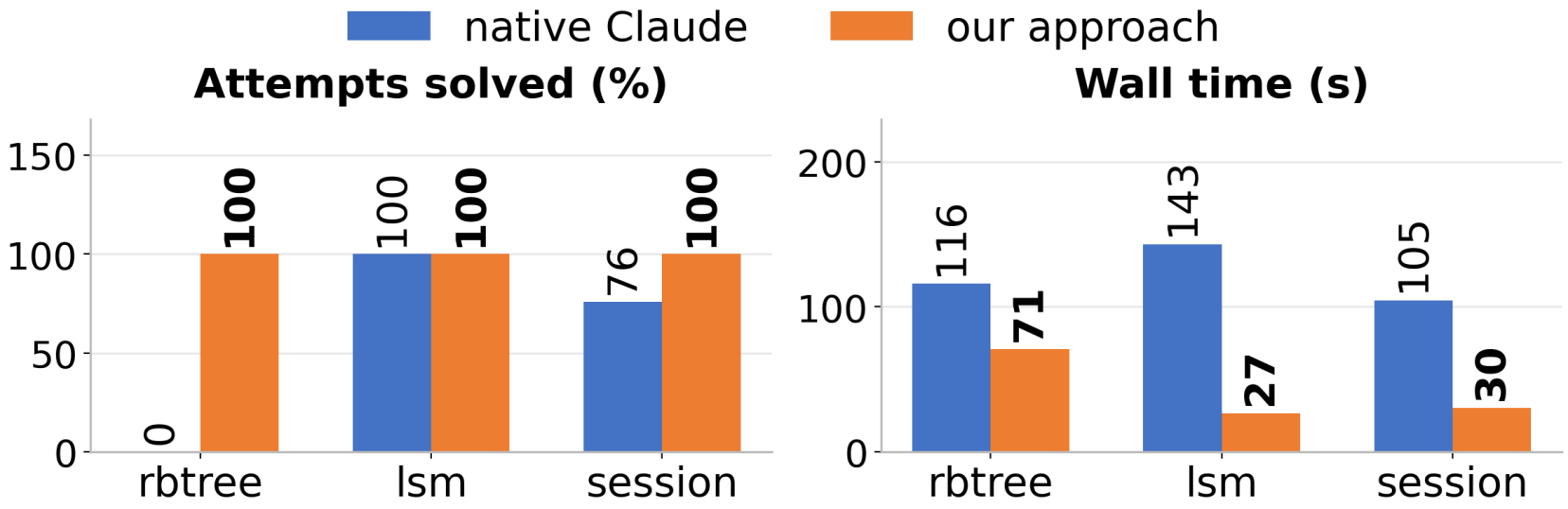
3 tasks · 25 iterations · protocol judge · tokens, cost, time

Fewer tokens, less spend, similar effort



3.0× fewer tokens · 1.4× less spend · same turns

Solves more, and finishes sooner



44/75 → 75/75 solved · 2.9× less wall time

Grounding beats effort

0 / 25

native Claude solved

25 / 25

With our approach

1.8×

fewer output tokens

native burns 10.3k tokens per attempt re-deriving an invariant it gets wrong; the our approach spends 5.8k implementing one it was given

Constraints buy predictability

2.1×

cheaper

5.4×

faster

5.2×

fewer output tokens

$0.25 – $0.99

native spread

$0.20 – $0.25

solution spread

variance shrinks further than the mean does

Validated lessons compound within a run

−29% turns

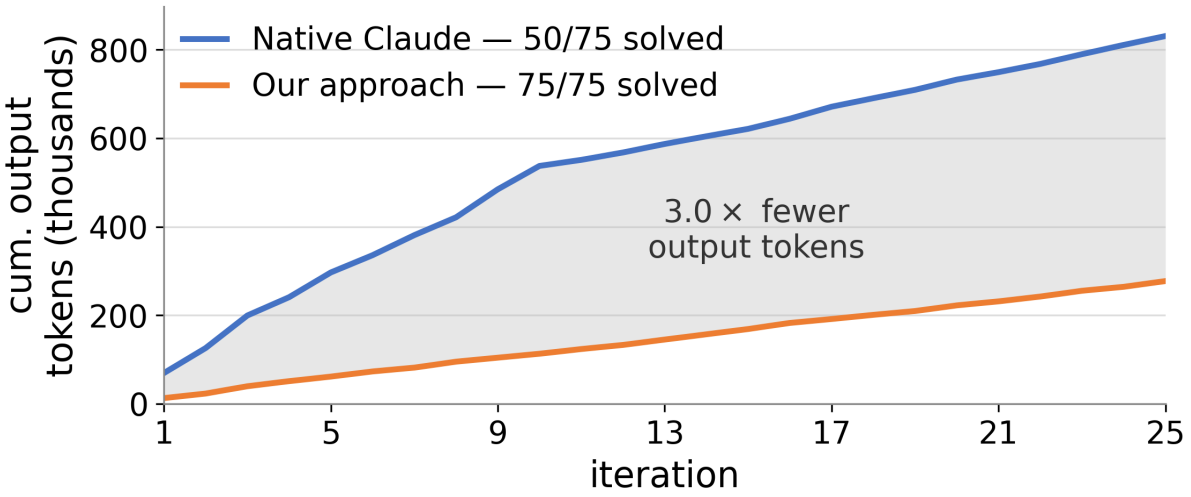
−20% cost

−26% wall time

first 5 vs last 5 iterations of the same run — ~40 lessons promoted, zero rollbacks

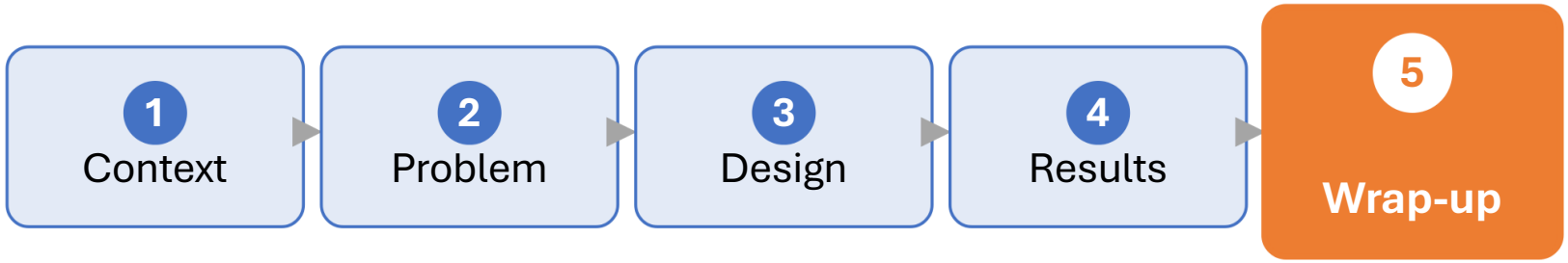
honest caveat: the curve is bimodal — lessons enable the fast path, they do not guarantee it

The gap widens on every iteration



831k → 277k output tokens · $30.17 → $21.26 · 50/75 → 75/75 solved

Next: what it all means



What to remember, and what comes next

Skills before smarts

OUR RECOMMENDATION

1 Context	2 Problem	3 Design	4 Results	5 Next
every stage	implicit	3 layers	75/75	auto-curate

KEY TAKEAWAYS

Grounding wins
0/25 to 25/25

Limits pay off
variance gone

Lessons stack
29% fewer turns

Thank You!

Skills before smarts
let's engineer the harness together