THE EVALUATOR ALSO VARIES: LOCALIZING NONDETERMINISM IN INTERACTIVE AGENT BENCHMARKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Interactive agent benchmarks can use large language models both as agents and as components of the evaluation system. When repeated evaluations disagree, the variation may originate in the simulated user or environment before it appears in the agent's behavior. We introduce a first-divergence audit that identifies the earliest disagreement between replicated interactions with identical preceding recorded histories, locating it in the agent, simulated user, or environment. Across 100 airline and retail tasks on τ^2-bench, five greedy vLLM replicates per task reveal transcript divergence on all tasks, but reward variation on only 20. The first discrepancy appears in the simulated user on 83 of the 100 tasks. In banking, disagreements persist in environment outputs after prefix caching is disabled on the agent and simulator. Comparisons of serving configurations reveal different patterns of residual divergence. A complementary ALFWorld experiment shows that preceding serving history can change trajectories without changing aggregate success, while cache resets restore agreement in the tested conditions. Together, these results show that stable benchmark rewards can conceal an unstable evaluation process. Reproducibility must therefore be assessed across the complete interaction, including the simulated user, environment, and serving protocol.

1 INTRODUCTION

Benchmarks for tool-using conversational agents replace the human customer with a language model (Yao et al., 2024; Barres et al., 2025). Increasingly the grader is a language model too (Zheng et al., 2023), and in some domains so is the environment the agent acts on. The system under test is therefore surrounded by other models that are not under test, and a reported score is a joint property of all of them. That such evaluations are noisy is established, and the standing advice is to replicate more (Bjarnason et al., 2026; Mustahsan et al., 2025; Reimers & Gurevych, 2017). Replication answers how much a score moves. It does not answer where the movement comes from, and in a loop with several models that is the question that decides whether the noise belongs to the agent at all.

We ask that question directly. Given several replicates of the same evaluation, run with every source of sampling switched off, which participant is the first to disagree? The answer is available from artifacts every benchmark already produces, and in most of our primary tasks it is the simulated user rather than the agent. On τ^2-bench with both models decoded greedily on a default vLLM deployment, all 50 airline and all 50 retail tasks produce divergent transcripts across five replicates, yet only 12 and 8 change reward. The first observable disagreement is in the simulated user's turn on 83 of the 100 tasks, 82 of them at its opening utterance, with the recorded preceding transcript identical in every case. On the one domain whose environment answers queries through a retrieval stage, disabling prefix caching on the served models leaves twelve divergent tasks, each first differing at an environment tool result. The evaluator varies before the agent does, and the score does not show it.

An interval over repeated runs is a legitimate description of end-to-end performance under a stated protocol. What it cannot say is where the variation it summarises first appeared, which in a loop with several models is what decides whether to fix the agent, the simulator, the environment or the

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

serving stack. Score replication alone cannot determine that; a measurement that says who moved first can.

One investigated source of this serving sensitivity is prefix caching, a serving-layer optimisation that is on by default in vLLM and SGLang and reuses cached attention state across requests. That prefix caching can change greedy outputs is documented in an issue report and in serving documentation (vLLM contributors, 2026; SGLang contributors, 2026), and a concurrent preprint reports that it changes agent trajectories on a tool-calling benchmark (Patodiya, 2026). We treat that base phenomenon as established and corroborated rather than as our contribution. Our contribution is what a single-model benchmark cannot see: an audit that attributes nondeterminism to the component at which it first becomes observable, evidence that this is most often the evaluator, and a comparison of what existing serving controls leave behind.

Contributions.

- **Scores conceal an unstable evaluation.** Reward is stable on 76 to 84% of tasks whose transcripts are not (§4.1), and a serving history that changes 12 of 20 ALFWorld trajectories leaves the pass rate untouched (§4.5). Repeating the reported number does not establish that the same interaction occurred.
- **A first-divergence audit.** We compare replicates only while their recorded histories agree and attribute the earliest disagreement to its author and message kind (§3.3). Normalising first-discrepancy counts by the turns still comparable is what makes them interpretable across participants. The audit identifies the participant and message kind at the first observable discrepancy; it does not identify the underlying mechanism or partition reward variance among components.
- **Localization.** First discrepancies concentrate in the simulated user's opening turn, hazard 0.556 against 0.093 on its later turns and 0.121 on the agent's; the agent's own first turn is 0.066, so this is not a generic first-request effect at each endpoint. Free-text agent turns have a pooled hazard of 0.205 against 0.046 for structured tool calls, a direction that holds in 15 of 17 evaluable arms with one tie and one small reversal. And in banking an environment-path residual persists at hazard 0.43 to 0.44 with prefix caching on or off (§4.2, §4.3).
- **What serving controls leave behind.** The tested vLLM batch-invariant configuration leaves no observed agent-first divergence while simulator-first discrepancies remain; verified request serialisation reproduces 48 of 50 transcripts at exactly zero reward variance; deterministic inference reproduces 20 of 20 with its cache retained; disabling the cache leaves only the banking environment-path residual (§4.4).
- **Serving history belongs in the protocol.** Varying only the traffic that precedes an ALFWorld episode changes 12 of 20 games; a requested cache reset before each episode, with the HTTP response recorded, restores agreement on 120 of 120 comparisons over the same 20 games (§4.5).

2 Related work

Noise in agentic evaluation. Bjarnason et al. (2026) find single-run pass rates on SWE-bench Verified (Jimenez et al., 2024) moving by 2.2 to 6.0 points between runs even at temperature zero, and also study where trajectories first diverge at the token level; Mustahsan et al. (2025) quantify the same instability with intraclass correlation. Both recommend replication. Random-effects decompositions split evaluation variance into named sources, crossed for language-model pipelines (Messing, 2026) and nested over scenario, trial and judge for a simulator-driven benchmark (Bogavelli et al., 2026), the latter naming the conflation of agent and simulator variance without resolving it. Our contribution relative to the first-divergence idea is to attribute the divergence to a participant and a message kind inside a multi-component evaluation loop, using only the transcripts a replicated run already produced.

Simulated users. Dou et al. (2025) score simulators by how well they track human judgements and Zhou et al. (2026) run the protocol with real users; those results bound what a simulator-based score can mean, ours bound how far it repeats.

Nondeterminism in serving. He & Thinking Machines Lab (2025) attribute nondeterministic outputs to batch-dependent kernel reductions and propose batch-invariant kernels; SGLang ships such kernels as a deterministic-inference mode designed to retain its radix cache (LMSYS Org & SGLang contributors, 2025). Pape et al. (2026) hold weights, decoding and hardware fixed across five engines and find backend choice alone shifts single-turn scores by up to 16.6 points, naming prefix caching among the drivers; Atil et al. (2024) document run-to-run disagreement under nominally deterministic settings. Closest to us, Patodiya (2026) reports, concurrently, that enabling prefix caching changes Berkeley Function Calling Leaderboard trajectories relative to a cache-off run in 36.2% of episodes at 16-bit precision, with 20.0% within-configuration repeated-run divergence, and that disabling it yields bit-identical runs in 800 of 800; Lumer et al. (2026) evaluate prompt caching for cost and latency without measuring output change. Those studies ask whether a single model's output moves. We ask which of several models in an evaluation loop moves first.

3 SETUP

3.1 BENCHMARK, MODELS, SERVING

We use τ^2-bench (Barres et al., 2025) at commit `363133a`, in which an agent and a simulated user act on a shared environment. Its domains differ in what the evaluator contains: `airline` (50 tasks) is scored programmatically; `retail` (114 tasks) carries a natural-language-assertion reward basis on 112, of which 40 have non-empty assertions and so can invoke a language-model judge (Zheng et al., 2023); among the 50 selected retail tasks the judge was invoked on 11 in condition A and on 18 distinct tasks across conditions A–D; `telecom`'s simulated user has tools of its own; and `banking`'s environment answers knowledge-base queries through lexical (BM25) retrieval over 698 documents, run in the benchmark's offline `bm25_grep` variant with no language model in the environment, so its environment can vary independently of either conversational participant and no serving control reaches it. The primary pairing is `Qwen3-30B-A3B` (Yang et al., 2025) as agent and `Mistral-7B-Instruct-v0.3` as simulated user; further pairings are in Appendix E. Reasoning is disabled, conversations are capped at 30 steps and calls at 1024 tokens, and the benchmark's silent retry of failed episodes is set to zero. Models are served through vLLM 0.11.0, with 0.11.2 for the batch-invariant and engine-version arms, and SGLang 0.5.2 and 0.5.3, with identical weights and one model per container, so the agent and the simulated user have separate caches. The retail judge calls the simulator endpoint, routed by a two-file patch to the benchmark that we supply (Appendix Q). One serving setting is varied at a time. Our second benchmark, ALFWorld (Shridhar et al., 2021), is a symbolic TextWorld environment in which the agent is the only language model present.

3.2 TRANSCRIPT IDENTITY, AND WHAT THE SCORE HIDES

Each message m is reduced to a signature $\sigma(m) = (\text{role}, \text{text}, \langle(\text{name}, \text{arguments})\rangle)$, tool-call arguments canonicalised, and ids, timestamps and token counts excluded. Given R replicate transcripts $s_1, \ldots, s_R$ of a task, extend each with $\tilde{\sigma}_r(i) = \sigma(s_r[i])$ for $i < |s_r|$ and $\perp$ otherwise; the *first divergence* is the least $i < \max_r |s_r|$ at which the $\tilde{\sigma}_r(i)$ are not all equal, so that unequal termination counts and identical finite transcripts do not. A task is *identical* when no such i exists. Equality throughout is equality of these transcript signatures, not of request payloads or rendered prompt tokens. Tool-result text is included, so a differing environment output is a divergence in its own right. Unequal termination counts as divergence; identical truncation at the same cap does not by itself, and termination reasons are recorded separately. Errored cells are dropped rather than counted as identical failure transcripts; a task enters a matched comparison with its usable replicates, of which it must have at least two in each arm.

3.3 THE FIRST-DIVERGENCE AUDIT

A count of where divergence first lands has an obvious confound: the simulated user speaks first, so it has the first opportunity to differ. The audit normalises for it (Figure 1). For each task with at least two replicates, every generated turn before the first divergence is a turn that was at risk and survived; the divergent turn is a turn that was at risk and diverged. Attributing each turn to its author and its kind, the ratio of events to turns at risk is a discrete-time hazard conditional on the

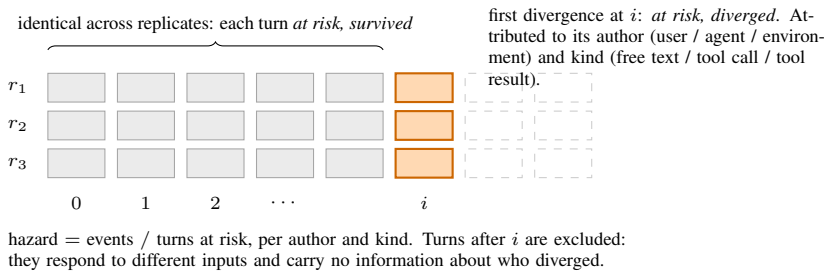


Figure 1: The audit. Replicates are compared only while their recorded histories agree. Every generated turn before the first divergence has survived under equal conditions; the divergent turn is attributed to whoever produced it. The resulting per-turn hazard compares participants only where each had the same opportunity, which normalises for the advantage of speaking first.

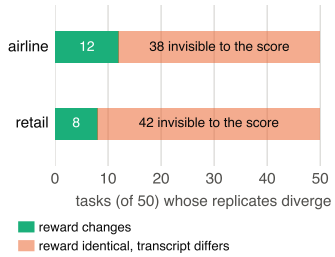


Figure 2: Every task diverges; a minority change reward. Airline and retail, 50 tasks each, five greedy replicates. The orange portion is instability the reported score cannot show.

entire preceding transcript being identical. Turns after the first divergence are excluded, since they respond to different inputs and say nothing about who diverged. Message kinds are read from the first replicate. The pooled estimates below use every cache-on τ^2-bench arm that has a recoverable serving record and is not itself a remedy arm, eighteen arms in all, mixing five-replicate and three-replicate designs; Appendix A lists them, gives every per-arm value, and reports the same quantities stratified by replicate count, since more replicates give more opportunity to observe a disagreement.

Three properties bound what the audit can show. It identifies the participant and message kind at the first *observable* discrepancy; the emitting component can be the first observable location even when the perturbation arose in shared serving state, request construction or retrieval, so the audit identifies neither the mechanism nor a partition of reward variance. Its comparisons across positions and roles are descriptive, since model, prompt, message length, per-endpoint cache history and survival to a later turn all differ between the turns compared, and the agent and simulator are served from separate containers with separate caches. And because later turns are excluded once replicates diverge, a hazard of zero means no participant-first discrepancy was observed among the turns still comparable, not that every request that participant made was reproducible. Nothing in §4 required an arm to be collected for this purpose, and a reader with stored replicates can run it on their own benchmark this afternoon.

4 Results

4.1 Reward stability does not establish a stable interaction

With both models at temperature 0 on a default vLLM deployment, 0 of 50 airline tasks and 0 of 50 retail tasks produce identical transcripts across five replicates. Reward changes on 12 and 8 of them (Figure 2). Between 76% and 84% of the non-reproducibility never reaches the number that gets reported. Reward is lossy for a structural reason: it is a coarse, usually binary function of a long trajectory, so two runs can take different paths to the same outcome. The concealment is not harmless. Each replicate is one complete single run, and on airline the five single-run pass rates are 0.200, 0.160, 0.160, 0.140 and 0.140, a six-point spread from a configuration a practitioner would call deterministic, comparable to the 2.2 to 6.0 points Bjarnason et al. (2026) report for an agentic benchmark. A practitioner who reports one of those runs has a one-in-five chance of reporting 0.200

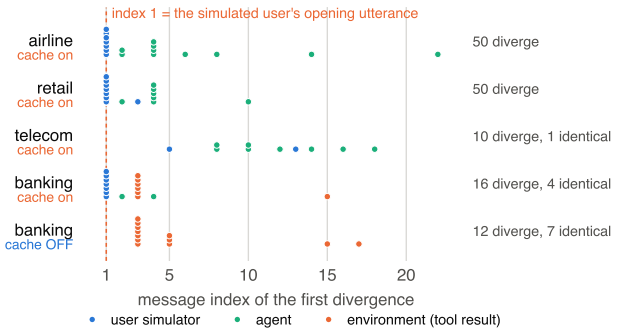


Figure 3: Where the first divergence lands, one point per task, coloured by whose turn it was. Under a default deployment the mass sits at message index 1, the simulated user's opening utterance. The bottom row is banking with the cache off: every surviving divergence is an environment tool result.

for a configuration whose five-run mean is 0.160. The same concealment appears on ALFWorld in §4.5, where a serving history that rewrites 12 of 20 trajectories leaves the pass rate at 0.100 throughout.

A verified serialised arm makes the distinction sharpest. With requests served one at a time and the configuration read back from the server's launch arguments (E26, 50 tasks by five replicates), pooled within-task reward variance in the greedy condition is exactly 0.000 and every replicate reports a pass rate of 0.120, yet 2 of 50 tasks still produce different transcripts, one first differing in the simulated user's turn at index 15 and one in the agent's at index 4, in both cases with the first replicate the odd one out (Appendix D). Stable rewards and residual trajectory differences coexist in the same arm.

4.2 Where the evaluation first stops reproducing

The raw count. On airline, 40 of 50 tasks first diverge at message index 1, the simulated user's opening utterance; on retail, 42 of 50 (Figure 3), plus one retail task first differing in the simulator's turn at index 3, for 83 simulator-first tasks in all. In all 82 opening-turn cases the recorded transcript before that message is identical across replicates. Twenty-eight airline and 26 retail tasks produce three or more distinct openings from five greedy runs, and three airline tasks produce five. The agent has not yet spoken; each replicate is handed a different version of the task.

The audit. Raw counts cannot say whether the simulator is fragile or merely first. The hazard can (Figure 4a). The simulated user's opening turn diverges with hazard 0.556, 227 events over 408 turns at risk. Its later turns diverge at 0.093, and the agent's later turns at 0.121. When the descriptive comparison is restricted to later turns, the simulator's hazard is not higher than the agent's. The concentration is specific to the simulator's opening utterance rather than to first requests in general: the agent's own first turn diverges at 0.066 (12 of 181), against 0.267 on its second and 0.097 thereafter. This is consistent with a cache-state effect on the simulator's opening request, but the analysis does not isolate position from the other differences listed in §3.3, and we report it as an observation rather than a mechanism. Under the tested batch-invariant configuration no agent-first discrepancy was observed among comparable turns while the simulator retained eleven, ten at its opening turn (§4.4).

What the opening utterance shares with no other turn is its input: its prefix, the simulator's instructions and the task, is identical across replicates and largely shared across tasks, so it is the request most likely to meet cache state left by an earlier episode. Testing that needs per-endpoint request ordinals and hit/miss records our arms did not log. What the arms establish is narrower and sufficient: before the agent has acted, the replicates already differ, in a component that belongs to the evaluator.

Output type. Restricting to agent turns after the first, so that the models and the position band are the same, free-text turns have a pooled hazard of 0.205 and structured tool calls 0.046, a 4.4-fold gap. The direction holds strictly in 15 of the 17 arms in which both kinds occur, with one tie at zero and one small reversal in the reasoning-enabled arm (Figure 4b, Appendix A). A natural hypothesis is that a chosen function call leaves a wider margin to the next-best token than free text does; we did

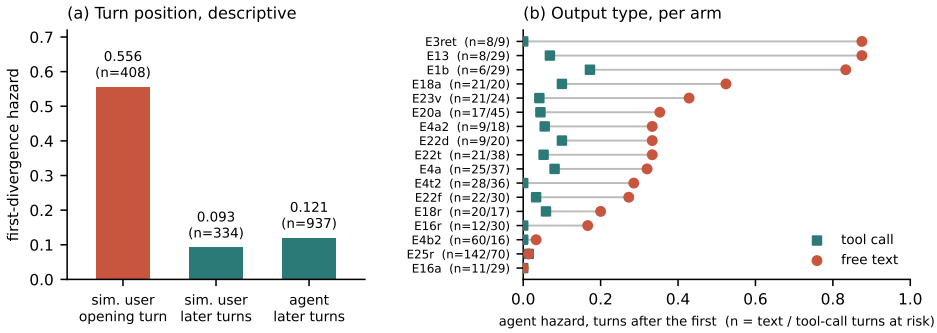


Figure 4: Two effects the audit separates. **(a)** Position. The simulated user's opening turn diverges at 0.556; its later turns and the agent's diverge at 0.093 and 0.121. **(b)** Output type, agent turns after the first, one row per arm with turns at risk shown: free-text turns have the higher hazard in 15 of 17 arms where both kinds occur, with one tie and one small reversal, pooling to 0.205 against 0.046.

not measure logit margins and state it as a hypothesis. Free-text turns are also typically longer, and length is not controlled. Whether output-type mix explains the lower divergence rate reported by the concurrent tool-calling study is plausible but untested here.

This is also why the audit stops at the first divergence. Once the opening utterance differs, every later agent turn answers a different input, and the 28 airline and 26 retail tasks with three or more distinct openings are, for the agent, that many distinct tasks under one id.

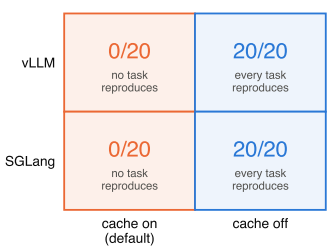
4.3 An environment-path residual that caching controls do not remove

Every domain except banking has an environment hazard of exactly 0.000 in every arm, cache on or off. Banking's is 0.438 with the cache on and 0.429 with it off. Disabling the cache moves banking's identity from 4 of 19 to 7 of 19, short of significance (McNemar exact $p = 0.375$), and the twelve tasks that still diverge all first diverge at an environment tool result, none at either model (Figure 3, bottom row). Tool use itself is not the explanation: airline, retail and telecom carry 375, 252 and 189 tool-result messages across three replicates in their cache-off arms, 125, 84 and 63 at-risk tool-result turns, and reproduce completely. What distinguishes banking is a retrieval stage in its environment: knowledge-base queries are answered by BM25 lexical retrieval, which no serving control reaches. In all 19 environment-first events across the two banking arms the returned score sequences are identical across replicates; in 17 the same documents appear in a different order and in 2 a different document with the same score crosses the top-k boundary, so the residual is the order among tied scores, which the benchmark does not fix (Appendix C). A per-role temperature design cannot attribute it. A benchmark whose environment or grader carries an unpinned source of its own, a tie order here or a language model elsewhere, carries a term that replicating the agent does not reduce.

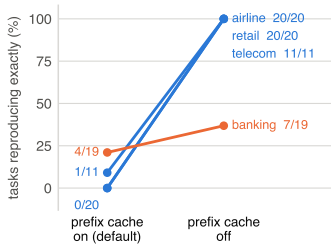
4.4 What serving controls leave behind

The base phenomenon. In a matched 2×2, no airline task reproduces with prefix caching on and every task reproduces with it off, on both vLLM and SGLang (Figure 5a). The same flag carries three agent architectures on airline and retail to complete repeatability, every discordant task running the same way (Appendix E). A single-request probe locates the mechanism in cache residency: a cache-miss request and a byte-identical cache-hit request produce different greedy output, and the miss output is invariant under unchunked prefill and a changed block size while the hit output is not (Appendix J). All of this is consistent with the concurrent report of Patodiya (2026) on a different benchmark, engine pair and model family, and we treat the two as independent corroboration.

Transfer to a benchmark with one model. ALFWorld removes the simulated user, the judge and the retrieval environment at once, so it tests whether the serving sensitivity needs any of them. It does not: identity moves from 0 of 20 games with the cache on to 20 of 20 with it off (McNemar exact $p = 1.9 \times 10^{-6}$), at a cache-on pass rate of 0.117 against a cache-off 0.100. With one language



(a) Engine × cache, matched airline tasks.



(b) The same flag, in each of four domains.

Figure 5: The base phenomenon. **(a)** Prefix-cache status predicts repeatability exactly on two independently implemented caches. **(b)** Each line is one domain over the tasks usable in both arms. Three reach complete reproducibility; banking, whose environment includes a retrieval stage with unpinned tie order, does not.

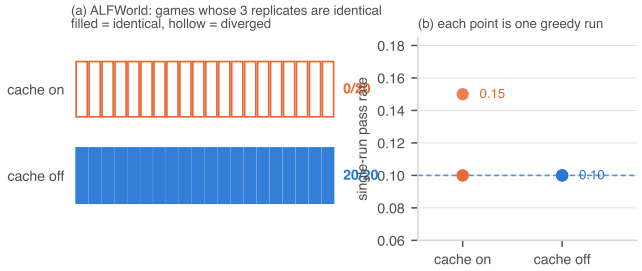


Figure 6: ALFWorld, 20 games by three replicates, primary agent, two-shot ReAct prompt. The agent is the only language model in the loop and a reset game returns a byte-identical observation. Every first-differing element in the cache-on arm is an agent action, as a deterministic environment requires, and the first divergence is late, at a mean of step 14; with the cache off every game reproduces.

model and a deterministic environment, every first-differing element is an agent action, as expected, and the mean first divergence is at step 14 of 50 with none at the first step (Figure 6). ALFWorld is transfer evidence that trajectories are serving-sensitive without an evaluator model in the loop; it is not a controlled reassignment of the first-speaker role and does not show the opening-turn concentration of §4.2.

What each control removes. Table 1 reads the audit against the remedies that exist. vLLM's batch-invariant mode (He & Thinking Machines Lab, 2025), verified active at a 97.5% prefix-cache hit rate, leaves no observed agent-first divergence in this arm while simulator-first discrepancies remain: 11 of 20 tasks still diverge, all first in the simulated user's turn, 10 of 11 at its opening utterance. The batch-invariance approach is designed to cover both request batching and the cached/fresh partitioning of a sequence; these results show only that the tested implementation retained simulator-first discrepancies, not why. Verified serialisation reaches 48 of 50, with replicate 0 the odd one out in both residual tasks, one first differing in the simulator's turn and one in the agent's. SGLang's deterministic mode reproduces 20 of 20 with the radix cache retained, against 2 of 20 for its baseline; both arms serve one request at a time, read live from each server, so batch composition is not what the mode repairs (Appendix J). Disabling the cache leaves no observed model-first discrepancy in any domain and leaves the banking environment-path residual in place. No single tested control reaches every participant, and an audit that examined only the agent would have declared the batch-invariant configuration a complete fix.

Two boundaries. Estimated directly as an interaction with a paired task bootstrap, the effect of cache state on which configuration ranks higher includes zero on the one simulator-matched agent pair and on both agent–simulator pairing comparisons we have (Appendix M); we make no claim that leaderboards flip. And exact repeatability is not accuracy. Two serving configurations that are each perfectly repeatable, differing only in attention backend with the cache off, agree on no task at all and report airline pass rates of 0.100 and 0.050 (Appendix G). Once replicates are bit-

Table 1: Residual divergence hazard by participant under each serving control, condition A, primary pairing on airline unless stated. Each control was collected as its own arm; these are observed patterns, not a controlled ranking. E26's configuration was read back from its launch log and E20a's and E21a's from each server's live settings; E4a's serialisation is a manifest constant with no surviving log. E20a and E21a both serve one request at a time. A zero is the absence of an observed participant-first discrepancy among comparable turns, not a demonstration that every request was reproducible.

Arm	Serving control	simulator	agent	environment	identity
E4a	cache on, serialisation asserted, unverified	0.243	0.177	0.000	0/20
E23b	batch-invariant kernels, cache on	0.138	**0.000**	0.000	9/20
E26	requests serialised, verified, cache on	0.003	0.002	0.000	48/50
E21a	SGLang deterministic mode, cache on	**0.000**	**0.000**	0.000	20/20
E12	prefix cache off	0.000	0.000	0.000	20/20
E4b2	banking, cache on	0.095	0.026	**0.438**	4/19
E12b	banking, cache off	0.000	0.000	**0.429**	7/19

Table 2: Transcript agreement across serving histories on ALFWorld, 20 games per condition, 100 episodes per cache state. Model, engine, flags, hardware, task set, prompts and decoding are fixed; only the traffic issued before each episode varies. Resets call the engine's documented endpoint and the HTTP response is recorded; vLLM 0.11.0 returns 200 on receipt without checking that clearing succeeded, so this is acknowledgement, not verified clearing. Comparisons reuse the same 20 games in one session per condition and are not independent trials.

	cache on	cache off
four reset conditions, pairwise (6 pairs)	120/120	120/120
any reset condition vs. no reset (4 pairs)	**32/80**	80/80
games with more than one distinct transcript	**12/20**	0/20
pass rate, every condition	0.100	0.100

identical, the error bar has to move to task sampling and to serving configuration, because it no longer measures anything else.

4.5 Serving history belongs in the evaluation protocol

The configuration-sensitivity literature studies choices a report can pin (Pape et al., 2026; Atil et al., 2024). Cache state is residue left by whatever ran before. We hold everything fixed on ALFWorld and vary only the traffic issued before each episode: a requested cache reset alone, a reset followed by five short unrelated completions, by thirty long ones, or by ten sharing the ReAct prefix (Yao et al., 2023), and no reset at all (Appendix K). The control is exact: with the cache off, all 200 cross-history comparisons are identical. With it on, the four reset conditions agree with one another on all 120 comparisons, while the condition that does not reset disagrees with each of them on 12 of 20 games (Table 2). The pass rate is 0.100 in every condition, so the benchmark's own metric shows none of it.

What ran before an evaluation can change its transcripts, and the transcript alone does not record it. But a reset before each episode is a protocol, and here it works; at the single-request level a probe reading the engine's cache counters finds the cold output differing from the warm on 9 of 20 prompts and from a reset-then-warm output on 0 of 20 (Appendix J). Serving history should be stated in the protocol the way temperature is, and a reset is a practical way to make it stateable in the tested setting.

The three conditions that reset and then issue filler traffic agree with the bare reset on every comparison, so within this design what matters is whether a reset was requested, not what was issued afterwards; we claim no dose–response relationship, and a larger working set or a smaller cache might reveal one.

5 DISCUSSION

The argument runs in one direction: scores conceal variation (§4.1), the audit locates it (§4.2), serving controls expose different residuals (§4.4), and those residuals fall on the evaluator as often as on the agent. Reproducibility of an interactive evaluation is a property of the loop and must be assessed on the loop. The audit is cheap: a comparison over replicates a study has already run, needing no new arms, only stored transcripts.

There is a consequence for variance decompositions on simulator-driven benchmarks. Our own pre-registered decomposition, run on the default deployment, returned a greedy-baseline variance of 0.058 that exceeded both the simulator-temperature contrast at 0.044 and the agent-temperature contrast at 0.040 (Appendix D). Re-collected with the cache off, the greedy-baseline term is exactly zero and the two temperature contrasts rise to 0.100 and 0.076. The ordering does not change, the simulator contrast being the larger point estimate in both regimes, but the gap between them widens from 0.004 to 0.024, and the magnitudes of both contrasts roughly double. These are temperature-associated changes in observed reward variance rather than identified latent components, since raising one participant's temperature also changes the other's inputs and which evaluator operations run; we label them as contrasts accordingly. The serving regime changes what such a decomposition returns, and the greedy-baseline term should be characterised and, where it can be, removed before the contrasts are interpreted.

Concretely, a replicated evaluation of an interactive agent should report, beside its identity rate: the first-message hazard; the free-text and tool-call hazards of each participant; the environment hazard, reported separately wherever the environment or grader has a source of its own, a language model or a retrieval stage, since a per-role temperature design folds it into the greedy-baseline term and does not identify it separately (Appendix D); the cache setting and request concurrency of every served model, read back from the server rather than asserted; and whether cache state was reset between episodes. The first three take a script and stored transcripts. The last two are what most of our own arms lacked.

6 LIMITATIONS

The audit identifies the first observable location of a discrepancy, not its cause, and a zero hazard for one participant is censored by earlier divergences elsewhere. It reads message kinds from the first replicate, does not control message length, pools five- and three-replicate arms (stratified in Appendix A) and per-model estimates across partners; its position and role comparisons are descriptive, the opening-turn concentration is consistent with a cache-state effect without isolating one, and it is post hoc. Arms are small: 20 tasks and three replicates for most contrasts, 20 games for ALFWorld and for the history experiment, and an interaction test whose null is weak evidence of absence. Only E20a, E21a and E26 have their serving configuration read back from the running server; earlier arms recorded serialisation as an asserted constant, so where they disagree with the verified arm, at 0 to 2 of 20 against 48 of 50, we treat the verified arm as authoritative and claim no measured decomposition between cache and scheduling. The cache-off, batched cell does not exist. History-experiment resets are acknowledged, not verified by telemetry, and its 120 comparisons reuse 20 games. Banking's residual is localised to the order of tied BM25 documents; its process-level cause was not manipulated. Models are 7B to 32B, reasoning is off in the main arms (Appendix N), and hosted APIs are out of scope.

7 CONCLUSION

In an interactive benchmark with a language-model user simulator and an environment of its own, a repeated evaluation measures the whole loop, and the loop varies before the agent does. The first observable disagreement concentrates in the simulated user's opening utterance, consistent with a cache-state effect on that request though not isolated as one; free text carries a higher pooled hazard than tool calls; an environment-path residual survives cache disabling in banking; and each tested serving control leaves a different residual that an agent-only audit would miss. Rewards conceal most of this. A first-divergence audit over transcripts a study already has makes it visible, and a stated serving protocol, including a reset, makes it reportable.

REPRODUCIBILITY STATEMENT

Every claim rests on per-cell artifacts we release. The audit is defined in §3.3 and implemented in a single script that regenerates every hazard in this paper from the raw records (Appendix A). Serving configurations are given as exact engine flags in Appendix Q, with resolved software versions and an explicit list of what was and was not recorded per arm in Appendix S. Every arm is listed in Appendix P with its design, cell count, losses and identity outcome. The transcript-identity definition is given in §3.2 and Appendix O; the variance estimator and bootstrap procedure are described in Appendix D; Appendix T separates pre-registered from post hoc analyses. The two-file patch to the benchmark (Appendix Q), the hazard and divergence scripts and the per-arm hazard records they produce are included in the supplementary material. The raw per-cell transcripts, the ALFWorld harness and the remaining figure scripts are released on publication, after the deployment credentials embedded in the raw records are removed. A bit-exact replication is not expected on different hardware; the directional results should reproduce from what is disclosed.

STATEMENT ON THE USE OF AI

Language models are components of the system studied: the agent, the simulated user and the retail judge are language models and are named with exact versions in §3; the banking environment is a lexical retrieval stage with no language model. Separately, AI coding assistants were used substantively in this work: to write and debug the experiment harness, serving configurations and analysis scripts; to compute and re-verify every reported number from the stored per-cell records; to draft and revise the manuscript text; and to prepare figures. AI tools were also used to help identify a small number of potentially relevant references; all cited sources were subsequently checked by the authors against the original papers. The research questions, experimental design, interpretation and claims are the authors', who reviewed all generated code and text and take full responsibility for the accuracy and integrity of the work.

REFERENCES

Berk Atil, Sarp Aykent, Alexa Chittams, Lisheng Fu, Rebecca J. Passonneau, Evan Radcliffe, Guru Rajan Rajagopal, Adam Sloan, Tomasz Tudrej, Ferhan Ture, Zhe Wu, Lixinyu Xu, and Breck Baldwin. Non-determinism of "deterministic" LLM settings, 2024. URL https://arxiv.org/abs/2408.04667.

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. τ^2-bench: Evaluating conversational agents in a dual-control environment, 2025. URL https://arxiv.org/abs/2506.07982.

Bjarni Haukur Bjarnason, André Silva, and Martin Monperrus. On randomness in agentic evals, 2026. URL https://arxiv.org/abs/2602.07150.

Tara Bogavelli, Gabrielle Gauthier Melançon, Katrina Stankiewicz, Oluwanifemi Bamgbose, Fanny Riols, Hoang H. Nguyen, Raghav Mehndiratta, Lindsay Devon Brin, Joseph Marinier, Hari Subramani, Anil Madamala, Sridhar Krishna Nemala, and Srinivas Sunkara. EVA-Bench: A new end-to-end framework for evaluating voice agents, 2026. URL https://arxiv.org/abs/2605.13841.

Yao Dou, Michel Galley, Baolin Peng, Chris Kedzie, Weixin Cai, Alan Ritter, Chris Quirk, Wei Xu, and Jianfeng Gao. SimulatorArena: Are user simulators reliable proxies for multi-turn evaluation of ai assistants? In *Proceedings of EMNLP*, 2025. arXiv:2510.05444.

Google DeepMind. gemma-3-12b-it (model card, unsloth mirror). https://huggingface.co/unsloth/gemma-3-12b-it, 2025. Gemma Terms of Use.

Google DeepMind. gemma-4-e4b-it (model card). https://huggingface.co/google/gemma-4-E4B-it, 2026. Apache License 2.0.

Horace He and Thinking Machines Lab. Defeating nondeterminism in LLM inference. Thinking Machines Lab, Connectionism blog, 2025. URL https://thinkingmachines.ai/blog/defeating-nondeterminism-in-llm-inference/.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. SWE-bench: Can language models resolve real-world GitHub issues? In *International Conference on Learning Representations (ICLR)*, 2024.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles (SOSP)*, 2023.

LMSYS Org and SGLang contributors. Towards deterministic inference in SGLang and reproducible RL training. LMSYS blog, 22 September 2025. `https://lmsys.org/blog/2025-09-22-sglang-deterministic/`, 2025. Batch-invariant attention and sampling kernels building on He & Thinking Machines Lab (2025); shipped as `--enable-deterministic-inference` in sglang 0.5.3. Accessed 2026-09-05.

Elias Lumer, Faheem Nizar, Akshaya Jangiti, Kevin Frank, Anmol Gulati, Mandar Phadate, and Vamse Kumar Subbiah. Don't break the cache: An evaluation of prompt caching for long-horizon agentic tasks. arXiv preprint arXiv:2601.06007, 2026. URL `https://arxiv.org/abs/2601.06007`.

Solomon Messing. Hidden measurement error in LLM pipelines distorts annotation, evaluation, and benchmarking, 2026. URL `https://arxiv.org/abs/2604.11581`.

Mistral AI. Mistral-7b-instruct-v0.3 (model card). `https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.3`, 2024. Apache License 2.0.

Mistral AI. Mistral-small-24b-instruct-2501 (model card). `https://huggingface.co/mistralai/Mistral-Small-24B-Instruct-2501`, 2025. Apache License 2.0.

Zairah Mustahsan, Abel Lim, Megna Anand, Saahil Jain, and Bryan McCann. Stochasticity in agentic evaluations: Quantifying inconsistency with intraclass correlation, 2025. URL `https://arxiv.org/abs/2512.06710`.

Nous Research. Hermes 3 - llama-3.1 8b (model card). `https://huggingface.co/NousResearch/Hermes-3-Llama-3.1-8B`, 2024. Llama 3.1 Community License.

David Pape, Jonathan Evertz, and Lea Schönherr. The silent hyperparameter: Quantifying the impact of inference backends on LLM reproducibility, 2026. URL `https://arxiv.org/abs/2605.19537`. Accessed 2026-08-29.

Aditi Patodiya. Same request, different answer: Quantization amplifies cache-induced divergence in LLM serving. arXiv preprint arXiv:2609.04748, 2026. URL `https://arxiv.org/abs/2609.04748`. Posted 4 September 2026; concurrent work.

Nils Reimers and Iryna Gurevych. Reporting score distributions makes a difference: Performance study of LSTM-networks for sequence tagging. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 338–348, 2017. doi: 10.18653/v1/D17-1035.

SGLang contributors. Troubleshooting and frequently asked questions: the results are not deterministic. SGLang documentation, 2026. URL `https://docs.sglang.io/references/faq.html`. States that prefix caching "can also dispatch to different kernels"; recommends `--disable-radix-cache` with one request at a time, and offers `--enable-deterministic-inference`. Accessed 2026-08-29.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations*, 2021. arXiv:2010.03768.

vLLM contributors. vllm v1 with prefix caching: first request differs from subsequent identical requests at temperature=0. vLLM issue #40896, opened 26 April 2026, 2026. URL `https://github.com/vllm-project/vllm/issues/40896`. vLLM 0.19.0, V1 engine, NVIDIA H100 with CUDA 12.8; `--no-enable-prefix-caching` restores agreement. Accessed 2026-08-29.

An Yang et al. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023. arXiv:2210.03629.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains, 2024. URL https://arxiv.org/abs/2406.12045.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2023.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E. Gonzalez, Clark Barrett, and Ying Sheng. SGLang: Efficient execution of structured language model programs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Xuhui Zhou, Weiwei Sun, Qianou Ma, Yiqing Xie, Jiarui Liu, Weihua Du, Sean Welleck, Yiming Yang, Graham Neubig, Sherry Tongshuang Wu, and Maarten Sap. Mind the sim2real gap in user simulation for agentic tasks, 2026. URL https://arxiv.org/abs/2603.11245.

A THE FIRST-DIVERGENCE AUDIT IN FULL

This appendix gives every number behind §4.2, the arm set the pooled estimates use, and the ablations a reader would want before trusting a post hoc analysis. All values regenerate from the per-cell records with `scripts/32_hazard.py`, under the same missing-data rule as every matched comparison in the paper: errored cells are dropped and a task enters with its usable replicates, at least two of them.

Definition. For each task with at least two usable replicates in condition A, the first divergence index i is found as in §3.2. Every generated message at index $1 \leq j < i$ is a turn at risk that survived; the message at $j = i$ is a turn at risk that diverged; for tasks that never diverge, every generated message is at risk and survived. A turn is attributed to its author (the simulated user, the agent, or the environment for tool results) and to its kind (free text, structured tool call, or tool result), both read from the first replicate. The hazard for any subset of turns is events divided by turns at risk. Position 1 is the first generated message of the episode, always the simulated user's opening utterance on τ^2-bench. The audit identifies the first observable location of a discrepancy, not its cause; a zero for a participant is censored by earlier divergences elsewhere.

Arm set. The pooled estimates use the eighteen cache-on τ^2-bench arms with a recoverable serving record that are not themselves remedy arms: `E1b`, `E3ret`, `E4a`, `E4a2`, `E4r`, `E4t2`, `E4b2`, `E13_sglang_cache`, `E16a_c2_cache`, `E16r_c2_cache`, `E18a_c4_cache`, `E18r_c4_cache`, `E20a_sgl053_fa3`, `E22d_dflt_cache`, `E22t_triton_cache`, `E22f_flashinfer_cache`, `E23v_vllm112_cache`, `E25r_reasoning_cache`. They span two engines, three attention backends, two engine versions and three agent/simulator pairings (C1, C2 and C4). The pilot series E5 to E8 is excluded because no manifest survives; the batch-invariant and serialised arms are reported separately in Table 1.

Table 3: Position and output type, pooled over the eighteen arms.

author	subset	events	at risk	hazard
simulated user	opening turn (index 1)	227	408	0.556
simulated user	later turns	31	334	0.093
agent	later turns	113	937	0.121
agent, later turns	free text	90	440	0.205
agent, later turns	tool call	23	497	0.046

Ablation: output type within each arm. Table 4 gives the agent's free-text and tool-call hazards on turns after the first, arm by arm. The direction holds strictly in 15 of the 17 arms in which both kinds occur, with a tie at zero in `E16a` and a reversal of 0.0141 against 0.0143 in the reasoning-enabled arm `E25r`.

Ablation: replicate count. The pooled arm set mixes two five-replicate factorials with sixteen three-replicate arms, and more replicates give more opportunity to observe a disagreement, so the pooled values describe that mixture rather than a replication-independent rate. The opening-versus-later simulator direction and the agent free-text-versus-tool-call direction persist in both strata. Later-turn role comparisons vary by stratum, and the $R = 5$ later-turn simulator estimate rests on only two turns at risk.

replicates	sim. opening	sim. later	agent later	agent free text	agent tool call
$R = 5$ (E1b, E3ret)	82/100 (0.820)	1/2	17/52 (0.327)	12/14 (0.857)	5/38 (0.132)
planned $R = 3$ (16 arms)	145/308 (0.471)	30/332 (0.090)	96/885 (0.108)	78/426 (0.183)	18/459 (0.039)

Ablation: each participant's own turn ordinal. Whether the opening concentration is a property of first requests in general or of the simulated user's opening utterance in particular is answerable from the same records. By each participant's own turn ordinal, pooled over the eighteen arms: simulated user 227/408 (0.556) on its first turn, 20/89 (0.225) on its second and 11/245 (0.045) thereafter; agent 12/181 (0.066) on its first turn, 43/161 (0.267) on its second and 58/595 (0.097) thereafter. The

Table 4: Agent divergence hazard by output type, turns after the first, per arm.

arm	free text	tool call
E1b	5/6 (0.833)	5/29 (0.172)
E3ret	7/8 (0.875)	0/9 (0.000)
E4a	8/25 (0.320)	3/37 (0.081)
E4a2	3/9 (0.333)	1/18 (0.056)
E4r	n/a	n/a
E4t2	8/28 (0.286)	0/36 (0.000)
E4b2	2/60 (0.033)	0/16 (0.000)
E13_sglang_cache	7/8 (0.875)	2/29 (0.069)
E16a_c2_cache	0/11 (0.000)	0/29 (0.000)
E16r_c2_cache	2/12 (0.167)	0/30 (0.000)
E18a_c4_cache	11/21 (0.524)	2/20 (0.100)
E18r_c4_cache	4/20 (0.200)	1/17 (0.059)
E20a_sgl053_fa3	6/17 (0.353)	2/45 (0.044)
E22d_dflt_cache	3/9 (0.333)	2/20 (0.100)
E22t_triton_cache	7/21 (0.333)	2/38 (0.053)
E22f_flashinfer_cache	6/22 (0.273)	1/30 (0.033)
E23v_vllm112_cache	9/21 (0.429)	1/24 (0.042)
E25r_reasoning_cache	2/142 (0.014)	1/70 (0.014)

agent's own first request is not elevated, so the concentration is specific to the simulated user's opening utterance. This is a descriptive comparison; it does not isolate position from the other role- and request-specific differences named in §3.3.

Ablation: the environment term everywhere. The environment (tool-result) hazard is exactly zero in every arm outside banking, cache on or off. The banking arms give E4b2 7/16 (0.438) and E12b_banking_nocache 12/28 (0.429). The cache-off arms of the other three domains give 0/125 (0.000), 0/84 (0.000), 0/63 (0.000) over their at-risk tool-result turns. This localises banking's residual to the retrieval path; Appendix C describes the tied-score differences.

Exploratory: per model. On free-text turns after the first, with a task-level bootstrap of 3,000 resamples, the models with at least forty turns at risk in the pooled arm set are:

model (role)	at risk	hazard	95% CI
NousResearch/Hermes-3-Llama-3.1-8B (assistant)	41	0.366	[0.250, 0.500]
Qwen/Qwen3-30B-A3B (assistant)	376	0.194	[0.143, 0.265]
mistralai/Mistral-7B-Instruct-v0.3 (user)	308	0.078	[0.046, 0.124]

Per-model estimates pool across arms with different partners and are reported as exploratory. We do not claim a model ordering from them.

What the audit does not control. Message length is not controlled, and free-text turns are typically longer than tool calls, so length may carry part of the output-type effect. Kinds are read from the first replicate; if replicates disagree on kind at the divergent turn, the first replicate's is used. The analysis was designed after the arms were collected and is post hoc.

B BENCHMARK AND DOMAINS IN FULL

We use τ^2-bench (Barres et al., 2025) at commit 363133a (v1.0.1–17), a dual-control benchmark in which an agent and a simulated user both act on a shared environment. Its four domains differ in how success is verified, which is what makes the cross-domain comparison informative. airline (50 tasks) has reward basis DB $\wedge$ COMMUNICATE: the database check is programmatic and the communication check is literal substring matching, so no language model participates in scoring. It carries the primary factorial. retail (114 tasks) has basis DB $\wedge$ NL-ASSERTION on 112 of them; 40 of those carry non-empty assertions and can invoke a language-model judge (Zheng et al., 2023). In our 50-task factorial the judge was invoked on 11 tasks in condition A and on 18 across the four conditions, so retail is a complementary contrast in which a minority of tasks also

involve a judge. The domains differ in tasks, policies and difficulty too, so this is a contrast, not an isolation of the judge. `telecom` is dual-control in the strong sense, its simulated user having tools of its own, so in principle a third participant can act. And `banking`'s environment answers knowledge-base queries through BM25 lexical retrieval over 698 documents, in the benchmark's offline `bm25_grep` variant, so no model call is made from the environment. It is the one domain whose environment can be nondeterministic independently of either conversational participant.

The primary configuration pairs `Qwen/Qwen3-30B-A3B` (mixture of experts, ≈3B active) (Yang et al., 2025) as agent with `mistralai/Mistral-7B-Instruct-v0.3` as simulated user. Three further pairings are used for the generalisation arms (Table 13); in every one the two roles are drawn from different model families, which reduces the risk that an observed effect is specific to one shared architecture. Reasoning is disabled except in the reasoning-scope arms of Appendix N (`enable_thinking: false`), being an additional source of stochasticity that would otherwise fold into the agent term. Conversations are capped at 30 steps and individual calls at 1024 tokens. Retail's NL-assertion judge calls the *simulator* endpoint at temperature 0, so in every configuration it is served by whichever model is acting as simulator; the banking environment makes no model call.

C THE BANKING RESIDUAL IN FULL

Figure 5b applies the same one-flag manipulation in each domain, over the tasks usable in both arms. Airline, retail and telecom all move from near zero to complete reproducibility. Banking moves from 4 of 19 to 7 of 19 and stops. That change rests on five discordant tasks, four improving and one regressing, and does not reach significance (McNemar exact $p = 0.375$); we report it as a direction the data is consistent with rather than as an effect. What the banking arms do establish is where the remainder sits, which does not depend on the size of the improvement.

What is left in banking is clean. All twelve remaining first divergences occur at a *tool result*, none at either model (Figure 3, bottom row). With caching off, no residual first divergence in any domain originated at either conversational participant. What survives is localised to the environment path.

Tool use alone does not explain it. Airline, retail and telecom all involve agent-issued tool calls, with 375, 252 and 189 tool results respectively in their cache-off arms, and all three reproduce completely. Banking differs from the other three in having a retrieval stage in its environment: the arms ran the benchmark's offline `bm25_grep` variant, in which the `KB_search` tool ranks 698 documents by BM25 score and the `grep` tool matches text, and no language model is called from the environment (the benchmark's optional model-backed interface class is never instantiated by this domain). Telecom is dual-control, so its simulator may call tools as well; in our runs it never did, every tool message in both telecom arms carrying `requestor: assistant`, so telecom cannot rule out simulator-side tool use specifically. Stated at the strength the data supports: tool use in general does not explain the residual, and what survives is localised to the environment path.

The environment-first events are all `KB_search` results: 12 in the cache-off arm and 7 in the cache-on arm. In every one of the 19 the returned score sequences agree across replicates, and the observed differences concern tied-score ordering or selection: in 17 the same documents are listed in a different order, and in 2 a different document with the same score is the one that crosses the top-k boundary. The residual is therefore the order among documents with tied scores. The retriever sorts by score with a stable sort, so that order is the order in which the document files were enumerated when the index was built, which the benchmark does not sort and our harness did not pin; each cell is a fresh process. A stable secondary sort key, or a pinned enumeration order, would test this directly, and we did not run that arm.

This is a distinct residual source of evaluation noise, and by Eq. (1) it is one the temperature factorial cannot attribute. The implication generalises past our setting: a benchmark whose environment or grader carries a source of its own, a retrieval tie order here or a language model elsewhere, can carry additional variance that a per-role temperature or seed design folds into its residual term, however many replicates are run.

D THE DECOMPOSITION ON BOTH SUBSTRATES, AND THE VERIFIED SERIALISED ARM

E26 protocol and result. E26 is the cell the two factorials leave open: prefix cache on, requests serialised. It ran 50 airline tasks by five replicates in condition A on the primary pairing, vLLM 0.11.0, with `-max-num-seqs 1`. The harness variable that sets it, `EVALPOWER_DETERMINISTIC`, adds only that flag to the serve command and changes no kernel or other serving setting, so E26 is serialisation alone. Its configuration was read back from the container's launch log, which records the serve command with `-max-num-seqs 1`, `enable_prefix_caching=True`, and prefix-cache hit rates between 54% and 95% during the run; it is the only vLLM cache-on arm in this paper with such a record. Pooled within-task reward variance in condition A is exactly 0.000 and every replicate reports a pass rate of 0.120. Transcript identity is 48 of 50. The two divergent tasks are task 17, first differing in the simulated user's turn at message index 15, and task 3, first differing in the agent's turn at index 4; in both the first replicate is the odd one out and the other four agree. Neither is an opening-utterance discrepancy.

The pre-registered manipulation is a temperature factorial: condition A holds both models at temperature 0, B samples only the simulator, C only the agent, D both. Writing s^2_{tX} for the unbiased within-task variance of reward over replicates of task t in condition X, and $\bar{V}_X$ for its mean over tasks with at least two usable replicates, the components are

$$\hat{V}_{\text{infra}} = \bar{V}_A, \quad \hat{V}_{\text{sim}} = \bar{V}_B - \bar{V}_A, \quad \hat{V}_{\text{agent}} = \bar{V}_C - \bar{V}_A, \quad \hat{V}_{\text{int}} = \bar{V}_D - \bar{V}_A - \hat{V}_{\text{sim}} - \hat{V}_{\text{agent}}. \qquad (1)$$

Components are never truncated at zero and the bootstrap resamples tasks, with 10^4 percentile resamples at a pre-registered seed. Both role terms are defined by manipulating the two conversational participants' temperatures, so the factorial does not separately identify environment or judge contributions: their settings stay fixed while their inputs, invocation patterns and contribution to observed reward variability can change across conditions, and whatever they contribute in the greedy condition lands inside $\hat{V}_{\text{infra}}$.

Table 5: The pre-registered decomposition of Eq. (1) on the two substrates, airline, $R = 5$, 50 tasks. Percentile bootstrap over tasks, 10^4 resamples. *identity* is transcript identity in condition A over the same 50 tasks. The columns are the greedy-baseline variance and the temperature contrasts of Eq. (1), not identified latent components.

substrate	$\hat{V}_{\text{infra}}$	$\hat{V}_{\text{sim}}$	$\hat{V}_{\text{agent}}$	$\hat{V}_{\text{int}}$	identity
cache on	0.058 [0.030, 0.090]	0.044 [0.016, 0.074]	0.040 [0.014, 0.070]	-0.034 [-0.074, 0.004]	0/50
cache off	**0.000** [0.000, 0.000]	0.100 [0.064, 0.138]	0.076 [0.044, 0.110]	-0.076 [-0.120, -0.036]	**50/50**

E PAIRING GENERALISATION IN FULL

The result above rests on one agent/simulator pairing. Because the cache-off arm is the paper's operational recommendation, we re-ran it on two further pairings, collecting *both* arms of each contrast fresh: 480 conversations across eight arms, every arm serialised at one request in flight, single replica, both models at temperature 0, 20 tasks $\times$ 3 replicates, with only the cache flag varied within a contrast.

Table 6 reports the result. Every pairing reproduces completely with the cache off and close to not at all with it on, and all 112 discordant tasks run in the same direction. This matters most for C2, whose agent is dense: C1's agent is a mixture of experts, whose routing is an independently known source of output nondeterminism, so cache and routing were least separable exactly where the original result leaned hardest. On the dense agent that confound is absent and the effect is unchanged.

Two caveats. Both C2 and C4 were run only on airline and retail, so the cross-domain boundary in §4.3, including banking's residual, still rests on C1 alone. And the cache-on arms here were recollected rather than taken from Table 13, because those earlier arms did not write a manifest and their serving configuration is not recoverable; the fresh C2 airline arm gives 2/20 where the earlier one gave 1/20, which is the kind of drift that reusing the published number would have hidden.

Table 6: Cache-off generalisation across agent/simulator pairings. Every arm collected fresh under one recorded configuration; only the prefix-cache flag differs within a row. Intervals are 95% Clopper-Pearson. p is McNemar's exact test, which is the paired test the design calls for since both arms measure the same task ids. C1 is the original study, restated here with the same tests.

Cfg	Agent	Domain	n	cache on	cache off	McNemar p
C1	Qwen3-30B-A3B (MoE)	airline	20	0/20 [0, 17]	20/20 [83, 100]	1.9×10^{-6}
C1	Qwen3-30B-A3B (MoE)	retail	20	0/20 [0, 17]	20/20 [83, 100]	1.9×10^{-6}
C2	Qwen3-32B (dense)	airline	20	2/20 [1, 32]	20/20 [83, 100]	7.6×10^{-6}
C2	Qwen3-32B (dense)	retail	20	0/20 [0, 17]	20/20 [83, 100]	1.9×10^{-6}
C4	Hermes-3-Llama-3.1-8B	airline	20	0/20 [0, 17]	20/20 [83, 100]	1.9×10^{-6}
C4	Hermes-3-Llama-3.1-8B	retail	16	2/16 [2, 38]	16/16 [79, 100]	1.2×10^{-4}

F LIMITATIONS IN FULL

The mechanism arms use 20 tasks and 3 replicates and, for most of them, one agent–simulator pair; the outcome is exact identity, an unusually sharp binary, so we treat 20 of 20 against 0 of 20 as decisive for direction while the arms stay small. The residency probe establishes cache residency as *a* coupling path and excludes prefill chunking and block size; it does not exclude allocator behaviour, and seven runs on three prompts with one model and one engine is a demonstration rather than a survey. The backend sweep covers three attention backends and two engine versions, and the batch-invariance result is on one engine's beta implementation, so "batch invariance is insufficient" is a statement about that implementation rather than about batch-invariant kernels in general; we could not exhibit the residual cache-state dependence in a single-request probe, because the target prefix stayed resident even after flooding, so the first-run signature is evidence for it rather than a direct demonstration.

The cache-off fix holds on three agent architectures, but only on airline and retail, so the cross-domain boundary, banking included, rests on one pairing, and within each pairing agent and simulator were not varied independently. Banking's residual is localised to the order among tied BM25 scores (Appendix C); the arm that pins that order, or replaces retrieval with gold documents, was not run. The banking null is itself under-powered at five discordant tasks, so "the fix does not reach the environment" is a direction the data are consistent with rather than a result they establish. The clean-substrate decomposition is an airline result, in a domain with no language model in its environment or grader, and Eq. (1) still folds such components into $\hat{V}_{\text{infra}}$ wherever they exist; the replicate-pair enumeration and the ICC were not repeated on the clean substrate.

The second benchmark is 20 games with one agent and one engine, so it shows transfer rather than surveying it, and the decision-relevance analysis is on 15–20 task subsets at three replicates, where the effect is proportionally larger than it would be on a full benchmark. The throughput measurement covers one model, one engine and concurrency up to 16 on a single accelerator, so a workload dominated by one very large shared prompt could still shift the balance towards the cache. The deterministic-mode result is on `sglang` 0.5.3 with `fa3` on H100s for both roles, its cost is a per-message proxy, and the mode bundles a sampling-backend change. Telecom's evidence is weaker than the other domains': its cache-off conversations are short, 33 of 60 cells ending within six messages against 3 of 60 on airline, and its cache-on arm covers 11 matched tasks rather than 20.

Every model is 7B–32B. A reasoning-enabled pair on the primary pairing preserves the direction, 17 of 20 against 20 of 20 with the same manipulation, but at three discordant tasks it does not establish significance, and the attenuation relative to the reasoning-disabled arms is confounded by an output budget that also changed (Appendix N); the strong form of our result is therefore a statement about non-reasoning configurations. Hosted APIs with provider-side prompt caching remain out of scope, and the gap matters: if the effect transfers there, the affected practitioners are precisely those who cannot act on it, and the burden shifts to providers to expose cache state and hit rates. Finally, the two benchmarks here are τ^2-bench and ALFWorld; Bjarnason et al. (2026) report early trajectory divergence on a third, which is consistent with a shared cause, but that study did not vary serving configuration.

Table 7: The two remedies, both on the primary pairing. *Fixed trace*: the twelve growing-prefix requests an agent issues over one real transcript, replayed three times against each deployment, warm passes reported, token counts from the server. *Per message* for the deterministic arms is conversation wall-time over message count, a proxy only, since the two arms produce different conversations when one diverges.

remedy	engine	cache	identical	cost measure	result
disable the cache	vLLM 0.11.0	off	20/20 (§4.4)	fixed trace, prompt tok/s	524.6 vs. 513.3 on (+2.2%)
deterministic mode	sglang 0.5.3	**on**	20/20 vs. 2/20	per message, s	4.64 vs. 1.23 ($\approx 3.8\times$)

G THE TWO REMEDIES IN FULL, AND TWO REPEATABLE CONFIGURATIONS THAT DISAGREE

Two repeatable configurations. E12 (vLLM 0.11.0, engine-default FlashAttention backend, cache off) and E24t (vLLM 0.11.0, Triton backend, cache off) are each perfectly repeatable, 20 of 20 tasks identical across their own three replicates. Compared with each other on the same 20 airline task ids, their transcripts agree on 0 of 20 tasks, reward differs on 3 of 20, and the airline pass rates are 0.100 and 0.050. Exact repeatability within a configuration does not bound the difference between two repeatable configurations. The throughput figures below are a fixed-trace measurement at one request in flight and a concurrency sweep to 16; "free" in the earlier text meant the fixed-trace timing difference, which was 2.2% in favour of cache-off at one request in flight, and should be read at that scope.

Disabling the cache shows no measurable cost at one request in flight on a fixed trace. Multi-turn evaluation re-sends the accumulated history on every turn, which is exactly the workload prefix reuse exists for, so the natural expectation is that disabling it costs a large multiple. Comparing conversation wall-clock between the existing arms cannot answer this, because those arms produce different conversations by construction. We instead replayed a fixed growing-prefix trace, the twelve requests an agent would issue over one real transcript, against the cache-on and cache-off deployments of the primary agent, three passes each so the cache is warm in the condition most favourable to it, and read token counts from the server. Cache-off is 2.2% *faster* on wall-clock (524.6 against 513.3 prompt tokens per second over 24 warm requests and 19,996 prompt tokens), so the cost is not merely small but not measurably positive; completion-token counts were identical in every pass of both conditions. This holds at one request in flight, the configuration this paper recommends, where there is little prefix reuse to lose. Because most harnesses run tasks concurrently, we repeated the measurement on a batching-capable deployment, one without `-max-num-seqs 1`, at three concurrency levels; at level C, C workers each replay the whole trace, so the server sees C conversations sharing a long common prefix, with a full warm-up pass first. Cache state is verified from the engine's counters rather than assumed: 219,956 queries at a 98.4% hit rate with the cache on, zero queries with it off. Disabling the cache costs 7.7% of total throughput at $C = 1$, 6.2% at $C = 4$ and 9.5% at $C = 16$ (Appendix H). The fear that a workload re-sending its history every turn would pay a several-fold penalty is not borne out: the cost is single-digit percent across a sixteen-fold range of concurrency, and reproducibility is cheap in either regime.

Keeping the cache with deterministic kernels works, and costs throughput. SGLang's deterministic-inference mode (LMSYS Org & SGLang contributors, 2025), built on batch-invariant kernels (He & Thinking Machines Lab, 2025), is intended to retain the radix cache. The flag does not exist in the sglang 0.5.2 used elsewhere in this paper, so we re-baselined on 0.5.3: two arms on the primary pairing, radix cache on, `fa3` attention, one request in flight, single replica, identical except for `-enable-deterministic-inference`. Both roles ran on H100s for this pair, because `fa3` is Hopper-only and in 0.5.3 only `fa3` keeps the radix cache under deterministic mode; the baseline shows the failure reproduces under that hardware change (2/20, eight of eighteen divergences at index 1), so the contrast is not confounded by it. With deterministic mode on, 20 of 20 tasks reproduce with the cache enabled (McNemar exact $p = 7.6 \times 10^{-6}$, 18 discordant tasks all one way), the first arm in this study to reach complete repeatability with a prefix cache on. The price is real: per-message wall time rises from 1.23 s to 4.64 s, about $3.8\times$. The mode also switches the sampling backend to PyTorch, so the contrast bundles two changes. The two remedies therefore trade oppositely: disabling the cache showed no measurable fixed-trace cost at one request in flight;

keeping it with deterministic kernels costs throughput. Which is cheaper depends on the concurrency regime, which this study does not measure.

H CONCURRENCY THROUGHPUT IN FULL

The same fixed growing-prefix trace as the serialised measurement, replayed against a batching-capable deployment (no `-max-num-seqs 1`) at three concurrency levels, with a full serial warm-up pass before timing so the cache is populated. Throughput is server-counted prompt plus completion tokens over wall-clock for the whole concurrent batch, so queueing is included rather than averaged away. All 192 requests succeeded at every level in both arms. Cache state is verified from the engine's counters: 219,956 prefix-cache queries against 216,432 hits with the cache on, and zero queries with it off.

concurrency	cache on (tok/s)	cache off (tok/s)	cost of disabling	p90 request (on / off)
1	573.4	529.0	−7.7%	1.83 s / 2.03 s
4	1970.3	1847.3	−6.2%	2.18 s / 2.39 s
16	6257.1	5665.6	−9.5%	2.86 s / 3.41 s

This is one model, one engine, one trace and concurrency up to 16 on a single accelerator, so it is not a general throughput law. Larger concurrency, longer shared prefixes, or a workload dominated by one very large system prompt would all shift the balance towards the cache.

I BATCH INVARIANCE IN DETAIL

Two arms, vLLM 0.11.2, default backend, prefix cache on, `-max-num-seqs 1` as the intended setting (asserted by the runner, not read back from the server; Appendix P), single replica, airline, 20 tasks × 3 replicates, 60 cells each, none lost. The only difference is `VLLM_BATCH_INVARIANT`. The cache is verified active in the batch-invariant arm from the engine's counters, 5.79M queries against 5.65M hits, a 97.5% hit rate, so this is not a cache-off arm under another name.

arm	batch inv.	cache	identical	95% CI	residual divergences
`E23v`	off	on	0/20	[0.0, 16.8]	10 agent, 10 simulator
`E23b`	**on**	on	**9/20**	[23.1, 68.5]	**0 agent**, 11 simulator
`E12` (ref.)	off	**off**	20/20	[83.2, 100.0]	n/a

Three observations locate the residual. It is entirely simulator-side and concentrated at the first generated message, 10 of the 11 residual divergences being at index 1. The input was byte-identical in all 11 cases; we compared the full message history preceding each divergence across replicates. And the odd replicate is the first run in 10 of 11 tasks, with replicates 1 and 2 agreeing, whereas in the baseline arm 14 of 20 divergent tasks have all three replicates differ.

A probe repeating one fixed prompt against the batch-invariant servers returns byte-identical output on every request, on both the agent (6/6) and the simulator (6/6), including the first cache-miss request; under default vLLM the same probe returns two distinct outputs. We attempted to extend it to *partial* residency, priming with a controlled fraction of the prompt, and could not: the KV cache on an H100 held the target prefix at 99.8% residency at every level, including after flooding with unrelated content. The residual is consistent with cache-state dependence and carries the first-run signature, though the request-level cache states were not measured for these residual events, and we cannot yet exhibit it in a single-request probe, and we report that as an open question.

J WHY DETERMINISTIC MODE FIXES IT AND BATCH INVARIANCE DOES NOT

Reproducible decoding under a prefix cache requires the reduction order to be independent both of which requests share a batch and of how much of a sequence arrived from cache rather than fresh prefill; the batch-invariance literature states both requirements (He & Thinking Machines Lab, 2025; LMSYS Org & SGLang contributors, 2025). What follows is an empirical audit of which

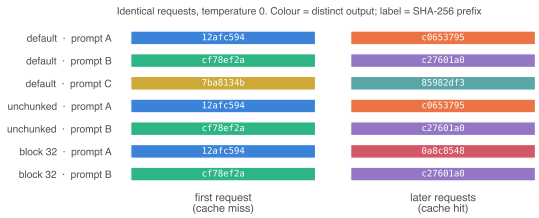


Figure 7: The miss/hit probe. Each row is 5 to 8 byte-identical requests of one prompt against the primary agent (vLLM 0.11, cache on, one request in flight, temperature 0, fixed seed); hit and miss status is read from the engine's own counters. Colour encodes a distinct output. In every row the first (miss) request disagrees with every later (hit) request. The miss column is unchanged by forcing prefill into a single chunk and by changing the KV block size; the hit column is not.

implementations meet which requirement. The evidence is three arms and one probe, and the load-bearing part is that in the decisive comparison the batch is already fixed at one.

The SGLang pair: batch size one in both arms. Both configurations were read live from each server's `/get_server_info` during the run, so they are measured rather than asserted.

Table 8: SGLang 0.5.3, both roles on H100, radix cache retained in both arms. Every field below was read from the running server. The two arms differ in the deterministic-inference flag, which additionally forces the sampling backend.

	`E20a`	`E21a`
`max_running_requests`	**1**	**1**
`page_size`	1	1
`chunked_prefill_size`	8192	8192
`attention_backend`	fa3	fa3
`disable_radix_cache`	False	False
`enable_deterministic_inference`	False	**True**
`sampling_backend`	flashinfer	pytorch
tasks reproducing exactly	**2/20**	**20/20**

Both arms serve one request at a time, so batch composition is constant in each and cannot be what differs; yet the non-deterministic arm fails on 18 of 20 tasks. Whatever deterministic mode repairs, it is not batching.

The same arm disposes of a tempting shortcut. `E20a` produces 13 tasks in which all three replicates differ, with the batch pinned to one, so a multi-valued divergence pattern is *not* evidence of batch composition. In a 31-message conversation residency is partial rather than binary, and once two replicates diverge at message k their later prefixes differ as well, so a cache alone can yield many distinct transcripts.

The residual under batch invariance alone. With `VLLM_BATCH_INVARIANT=1` and the cache retained, verified active at a 97.5% hit rate, identity moves from 0/20 to 9/20 and the surviving divergences change character: 10 of 11 have replicate 0 as the odd one out, all 11 are simulator-side, 10 of 11 begin at message index 1, and the message history preceding the divergence is byte-identical in 11 of 11. This first-run pattern is consistent with cache-state dependence, but request-level cache states were not recorded for these residual events, so their cause remains unestablished; the tested implementation did not remove it. The batch-invariance approach is designed to make the arithmetic independent both of which requests share a batch and of how a sequence is partitioned between cached and fresh computation (He & Thinking Machines Lab, 2025); whether the residual reflects incomplete kernel coverage in this release, another serving component, or a different source of variation is not established here.

The reset probe. Twenty prompts drawn from real airline transcripts (489 to several thousand tokens) were each issued to the primary agent endpoint in four states: cold, after a call to `/reset_prefix_cache`; warm, a byte-identical repeat; partial, after priming half the prompt; and reset-then-warm, a reset followed by a repeat. The engine's `prefix_cache_queries` and

`prefix_cache_hits` counters were read around every request, so hit and miss are measured, and the reset's effect is visible in the counters rather than inferred from its HTTP response. The cold output differed from the warm output on 9 of 20 prompts and from the reset-then-warm output on 0 of 20. This is the single-request counterpart of the history result, and it is the only place in this paper where a reset is verified rather than acknowledged.

The probe isolates the split directly. The probe issues single requests with one in flight, so no other request is co-batched with it. Forcing prefill into a single chunk leaves both outputs byte-identical to default chunking, so chunking of the *new* tokens is not the variable. Changing the KV block size leaves the *miss* output unchanged across all seven runs while changing the *hit* output's hash, with the hit count moving by exactly one block's granularity. The miss path prefills the whole sequence and is stable under every knob varied; the hit path's result depends on the cache's block layout, which is to say on where the cached/fresh boundary falls.

Reading. Within the tested implementations, disabling the cache and SGLang's deterministic mode reproduce every task; vLLM's batch-invariant mode reproduces 9 of 20; serialisation alone reproduces 2 of 20 on SGLang and 48 of 50 on the verified vLLM arm. Which invariance each implementation provides was not measured and is not tabulated.

What is inference rather than measurement. We measure that deterministic mode fixes the failure with the cache retained and that batch invariance alone does not. We do not measure SGLang's kernel internals, and the `E20a`-to-`E21a` contrast bundles deterministic attention with a sampling-backend change from flashinfer to pytorch, which Table 8 records. Attributing the fix specifically to split-invariant attention is therefore the best-supported reading of these data rather than a measurement of it. The experiment that would settle it runs with the cache *off*, so residency plays no role, and forces the prefill to split at k tokens while sweeping k across block boundaries, recording the top-2 logit margin at the first token that changes. The prediction, fixed in advance: outputs change only where the split crosses a boundary, and the tokens that flip are those whose margin lies below the accumulated floating-point error.

K HISTORY DEPENDENCE IN FULL

Protocol for §4.5. Agent `Qwen/Qwen3-30B-A3B` on an H100, vLLM 0.11.0 V1, `-max-num-seqs 1`, single replica, temperature 0, seed 0, reasoning disabled, 100 tokens per action, up to 50 environment steps, the standard two-shot ReAct prompt for each game's task type. Twenty evenly spaced games of the sorted `valid_unseen` split, identical across conditions and spanning all six task types. Each condition runs all twenty games sequentially in a single container, so episodes share a server and an order, which is the point: the cache carries state between them.

The five histories are *fresh* (documented reset, then straight into the episode), *light* (reset, then five 40-token unrelated completions), *heavy* (reset, then thirty 400-token ones), *shared* (reset, then ten 60-token completions that reuse the ReAct instruction opening, so the residue overlaps what the evaluation will ask for), and *natural* (no reset). Resets call the engine's documented endpoint (`/reset_prefix_cache` on vLLM, `/flush_cache` on SGLang) and the status code is recorded per episode; a condition whose reset did not return 200 is not counted as an independent history. Cache-off servers have no cache to reset and report `none`, which is expected.

Table 9: Transcript agreement across history conditions, same game, same configuration, ALFWorld, 20 games per condition, 100 episodes per cache state. Each entry counts games whose transcripts are identical between the two histories.

history pair	cache **on**	cache **off**
the four reset conditions, pairwise (6 pairs)	120/120	120/120
any reset condition vs. *natural* (4 pairs)	**32/80**	80/80
all pairs	152/200	**200/200**
games with more than one distinct transcript	**12/20**	0/20
pass rate, every condition	0.100	0.100
mean steps, every condition	46.2–46.7	47.1

Table 9 gives the counts. Three points of reading. The cache-off row is a control on the whole instrument, not only on the cache: it holds the harness, the environment, the prompt construction and the decoding to exact determinism across 200 comparisons, so any disagreement in the cache-on arm is attributable to the one thing that differs. The reset conditions agreeing with each other on 120 of 120 is consistent with a requested reset producing a comparable starting state, though the reset is acknowledged by HTTP status rather than verified by cache telemetry; on that reading *natural* is the one differently-prepared condition rather than treating all five as arbitrary. And the pass rate and mean step count are flat across every condition in both arms, so the effect is invisible to the benchmark's own metric and to any coarse proxy for effort.

We do not claim a dose–response relationship. The tested filler conditions were indistinguishable from a bare reset request. The history experiment did not record target-prefix residency, so it does not establish why filler traffic had no observed effect; a separate single-request probe found 99.8% residency after flooding at every level tried, which is consistent with a large cache but was not measured here.

L SECOND BENCHMARK: PROTOCOL AND RESULT

Both arms use the primary agent (`Qwen3-30B-A3B`, vLLM 0.11.0, one request in flight, single replica, temperature 0), the standard two-shot ReAct prompt for the game's task type taken unmodified from the ReAct repository, up to 50 environment steps, 100 output tokens per action, and 20 games evenly spaced through the sorted `valid_unseen` split so the set spans all six task types (put 4, clean 4, cool 4, heat 3, examine 3, puttwo 2). Three replicates per game, 60 cells per arm, none lost. A transcript is the sequence of (action, observation) pairs and identity is elementwise agreement, the same definition used for τ^2 transcripts. The arms differ only in `–no-enable-prefix-caching`.

Figure 6 in the main text shows the per-game outcome.

Counting actions from one, the first differing action in the cache-on arm has mean and median 14 of 50 and minimum 2, so no game differs at its first action and three differ by the second (an earlier draft reported the same records zero-indexed, as mean 13), and episodes are long enough for divergence to accumulate (mean 46.2 steps cache-on against 47.1 cache-off).

M THE COMPARISON RECOMPUTED ON BOTH SUBSTRATES

The quantity of interest is the interaction, the change in a pairwise gap between substrates, estimated on the joint paired data with a task-level bootstrap of 10^4 resamples over the tasks usable in both substrates. Only the C1–C2 pair holds the simulator fixed and is an agent comparison; the two pairs involving C4 change the simulator as well and are agent–simulator pairing comparisons.

pair	kind	gap, cache on	gap, cache off	interaction (off − on)
Qwen3-30B-A3B vs Qwen3-32B	agent	$+0.117\ [0.000, 0.267]$	$+0.050\ [0.000, 0.150]$	$-0.067\ [-0.233, 0.100]$
Qwen3-30B-A3B vs Hermes-3-8B	pairing	$-0.083\ [-0.267, 0.083]$	$-0.200\ [-0.400, -0.050]$	$-0.117\ [-0.383, 0.117]$
Qwen3-32B vs Hermes-3-8B	pairing	$-0.200\ [-0.400, 0.000]$	$-0.250\ [-0.450, -0.100]$	$-0.050\ [-0.283, 0.183]$

Table 10: Pairwise mean-reward gaps on each substrate and the direct interaction, airline, $n = 20$ tasks. All three interactions include zero. A gap that excludes zero on one substrate and includes it on the other is not evidence of an interaction; the last column is the test.

The comparison is airline, 20 shared tasks, 3 replicates; the per-substrate gaps and their intervals in Table 10 come from the same joint task bootstrap as the interaction column.

On retail the three cache-on replicates yield two distinct orderings of the three agent–simulator configurations and the cache-off replicates yield one.

N REASONING-ENABLED SCOPE ARM

Two arms identical except for the cache flag, with the agent's reasoning enabled and the output budget raised from 1024 to 4096 tokens so traces are not truncated mid-thought. Reasoning is

verified present rather than assumed: 87 of 93 sampled assistant messages carry a reasoning trace, mean length 2,368 characters, against 0 of 96 in the non-reasoning arm.

arm	reasoning	cache	identical	95% CI
`E25r`	on	on	17/20	[62.1, 96.8]
`E25n`	on	off	20/20	[83.2, 100.0]
`E4a` (ref.)	off	on	0/20	[0.0, 16.8]
`E12` (ref.)	off	off	20/20	[83.2, 100.0]

Within the reasoning-enabled pair the direction is unchanged and all three discordant tasks run the same way, but with only 3 discordant tasks McNemar gives $p = 0.25$: this arm does not establish significance. The three residual divergences are at message index 2 and at the agent, where the non-reasoning arm's were dominated by the simulator at index 1. The reference cache-on arm `E4a` is the one whose serialisation is asserted rather than verified (Table 1); the reasoning arms share that status.

The attenuation is real and its cause is confounded. With reasoning off the cache-on arm reproduces 0 of 20; with reasoning on it reproduces 17 of 20. We do not attribute that to reasoning, because these arms also raise the output budget, which changes generation for the simulator as well as the agent; isolating reasoning would need a third arm with reasoning off and the larger budget, which we did not run. We also did not record cache-hit rates for these arms, so the plausible mechanism, that long reasoning traces make each conversation's prefix diverge earlier and so reduce sharing, remains a hypothesis. The reasoning-enabled, larger-budget configuration has approximately twice the conversation wall-time: mean 284.5 s against 139.0 s at essentially unchanged length.

O Outcome measure, serving verification, replication cost, and controls

Four parts of the experimental account that the main text refers to but does not have room for, reproduced here unchanged.

Outcome measure

The pre-registered outcome was reward variance. After seeing the first results we added a stricter one, *transcript identity*. Each message m is reduced to the signature

$$\sigma(m) = \big(\text{role}(m),\ \text{text}(m),\ \langle(\text{name}, \text{arguments})\rangle_{\text{tool calls of } m}\big),$$

with tool-call arguments canonicalised by sorting keys. Given R replicate transcripts $s_1, \ldots, s_R$ for a task, the first divergence is the least index $i < \max_r |s_r|$ at which the extended signatures $\tilde{\sigma}_r(i)$ of §3.2, with $\perp$ past a transcript's end, are not all equal; equivalently, either some s_r has ended while another continues, or $|\{\sigma(s_r[i])\}_r| > 1$; the task is *identical* when no such i exists. We record the identity rate over tasks, the role at the first divergence, and the divergence index.

Two details of σ decide what the measure sees. It *excludes* the message id, tool-call id, timestamp, token-usage and latency fields, all of which differ between any two runs; including them would make the measure vacuous. It *includes* tool-result text, because a differing tool result means the environment diverged, which is a distinct finding from a model diverging. That inclusion is what isolates banking's residual in §4.3. Unequal termination counts as divergence for the same reason: stopping early is a behavioural difference. Identical truncation at the same cap does not itself constitute disagreement; termination reasons are recorded separately.

Serving, and how the configuration was verified

Two harness properties matter for correctness. First, τ^2-bench retries a failed simulation up to three times by default, which silently discards unlucky rollouts and causes the benchmark to underreport its own variance; we set retries to zero. Second, seeds are not a usable control here: on one endpoint we tested (SGLang serving `google/gemma-4-E4B-it` at T=1) `temperature` and

`top_p` were honoured while `seed` was accepted and then ignored. The design therefore anchors on temperature.

We serve identical weights through vLLM (Kwon et al., 2023) and SGLang (Zheng et al., 2024), varying one setting at a time (Appendix Q; versions in Appendix S). Requests were intended to be serialised in both, one in flight at a time, which is not the engine default. That setting is verified for `E26`, `E20a` and `E21a` and asserted for the remaining arms, and where the two disagree §4.4 and §6 treat the verified arm as authoritative. The primary temperature factorial used the deployment default; all later mechanism and cache-on/cache-off domain arms were configured to serialise requests, so that cache status could be studied without concurrent batch composition varying alongside it. Not every arm recorded that setting in a per-arm manifest, a logging gap the recommendation in §5 is partly aimed at; the later arms write a manifest recording the resolved model ids and the cache setting read back from the cells rather than asserted, and `E26` additionally records the running server's launch arguments (Appendix P). Models run one per container, the 30B agent on an H100 and the 7B simulator on an L40S, except in the deterministic-mode arms of §4.4, where the simulator also runs on an H100 for a reason given there.

Configuration here is the independent variable, so it should be verified rather than assumed; §6 states for which arms that was achieved. Flags are set from image-level environment variables rather than the launching shell, after an arm was lost to a variable set on the client and unset inside the container: vLLM ran with batching enabled while the deploy output looked correct, and that arm would have produced a confident null result. Three levels of configuration evidence should be distinguished: the intended setting, the setting a manifest recorded, and the setting read back from the running server. Until the arms of §4.4 the manifest recorded serialisation as a constant, so only the third level is verification. Before every arm the served model id was read from the live endpoint; the launch command or live settings were read back only for the three arms named in §6. For the remaining arms the served model identity was checked, but serialisation remained an intended or manifest-recorded setting rather than one independently verified from a retained launch log or live server arguments.

What replication level costs

Two further measurements sharpen the practical cost. First, at two replicates per task the estimator cannot order the two components practitioners compare. Enumerating all ten ways of choosing two replicates from five and applying each globally, the simulator term is larger in four draws, the agent term in four, and they tie in two, the difference spanning -0.040 to $+0.100$; at $R = 5$ on the same runs the two are indistinguishable (0.044 against 0.040). Which pair one happens to run decides the answer, and nothing in a single $R = 2$ result signals that. One instance is ours: the trial-order pair in Table 14 is the single draw of ten putting the agent estimate below zero. In this airline configuration two replicates were insufficient for a stable component ordering and estimates settled by four to five; we report that as a property of this configuration rather than a general rule.

Second, the share of variance lying between tasks, which is what a benchmark exists to measure, falls monotonically as stochasticity is added, from 0.574 with both models greedy to 0.202 with both sampling (Table 15). Under condition D, with both conversational models at temperature 1, four fifths of the observed variation no longer separates one task from another. We report this as a property of these 50 tasks under this configuration, not a general constant.

Both tables were computed on the cache-on substrate, as was the pre-registered decomposition (Appendix D); §5 and Appendix D repeat that decomposition with caching disabled. The replicate-pair enumeration and the ICC were not repeated there, so the two numbers above remain properties of a default deployment.

Controls: what the effect is not

The same mechanism subsumes a factor we had previously reported as independent. Replica count is not an independent mechanism. On vLLM with caching on, trajectories diverge roughly four times later when all requests are routed to a single serving replica (median normalised position 0.125 against 0.032), but the cache-off SGLang controls show that effect is cache-mediated in these arms, since single-replica and multi-replica arms both reproduce 20 of 20. Replica count is therefore

one more way cache state can differ between runs, which makes it an unreported evaluation variable whenever caching is left on, and it is on by default. We did not repeat the topology control on vLLM, so engine-specific scheduling effects are not ruled out more generally.

Several explanations are easier to reach for than the cache, so each was tested.

Not sampling. Every mechanism, cache and serving-control arm used to establish the cache effect decoded greedily on both sides, verified per cell from the recorded call arguments rather than from the intended configuration. Conditions B–D sample by design and are not part of that evidence. *Not seeds.* Accepted and ignored on the endpoint we tested (above, under serving); we did not re-test `seed` on the stacks serving the main arms, so that observation is scoped to that endpoint.

Batch composition is not separated from the cache. Batch composition is the usual suspect for order-dependent numerics (He & Thinking Machines Lab, 2025), and this study does not rule it out on vLLM. The arms whose serialisation is asserted rather than verified reproduce 0 to 2 of 20 airline tasks with the cache on, while the verified serialised arm reproduces 48 of 50 (§4.4); on SGLang, where both arms serve one request at a time by live server settings, the cache-on arm still fails on 18 of 20 tasks. The statement the data support is narrower: the cache-off arms reproduce completely under every scheduling we ran, serialised or autoscaled, and the cache-on arms do not.

Not the model family. With the cache on and the deployment default, no pairing in Table 13 reproduces more than 40% of its tasks, across three agent architectures and three simulators. The complementary statement now also holds: with the cache off, three architectures reproduce completely (§4.4).

Not engine identity within the two-engine factorial. Figure 5a shows both engines failing with their caches on and both succeeding with them off.

Not prefill chunking, and not block size. With the cache on, forcing the whole prompt into one prefill chunk leaves the miss-versus-hit difference unchanged, and so does changing the KV block size; the miss output is invariant under both (§4.4).

One contrast inside that table points back at the study's original question: with the Hermes agent held fixed, changing the simulator from Qwen3-8B to gemma-3-12b-it changes the observed repeatability and the first-discrepancy pattern (Appendix Q). These arms do not establish that simulator changes have larger effects than agent changes, since the other pairings change both roles at once. The clean-substrate decomposition gives the simulator term as the larger point estimate with the cache off (Appendix D), with the qualification stated there.

P COMPLETE ARM INVENTORY

Generalisation arms are in Table 13. Checks run on the five mechanism arms before believing the result: within each arm all 60 cells carry distinct seeds, so a task's replicates are separate simulations rather than one result copied (seeds are derived from task and trial index, so matched arms share the same grid); conversations are substantial, with a median of 31 messages; both reward values occur in every arm; and transcripts differ across tasks, so the comparator is not trivially reporting equality.

Pairing-generalisation arms. The eight arms behind Table 6 were collected separately from the main factorial, in order to test the recommendation on more than one pairing. Each is 20 tasks $\times$ 3 replicates, condition A, serialised at one request in flight, single replica, both models at temperature 0.

Arm	Cfg	Cache	Domain	cells	bad	ident.
`E15a`	C2	off	airline	60	0	20/20
`E15r`	C2	off	retail	60	0	20/20
`E16a`	C2	**on**	airline	60	0	2/20
`E16r`	C2	**on**	retail	60	0	0/20
`E17a`	C4	off	airline	60	0	20/20
`E17r`	C4	off	retail	60	9	17/17
`E18a`	C4	**on**	airline	60	0	0/20
`E18r`	C4	**on**	retail	60	10	2/17

Table 11: Every condition-A arm reaching the paper. *Serialised* is whether one request was in flight at a time (`max-num-seqs 1` / `max-running-requests 1`), which is *not* the engine default. It is verified only for `E26` (launch log, ‡ in the next table) and `E20a`/`E21a` (live server settings); for every other arm marked yes it is the intended setting, which the runner wrote to the manifest as a constant, so no batching-versus-cache comparison inside these groups is measured (§6). † manifest written for the arm; the remaining arms were configured identically but did not write one, which is a logging gap of ours (§3). *cells* is conversations dispatched, *bad* cells producing no usable transcript, *ident.* tasks whose replicates are identical message for message over that arm's own usable tasks, *pos* the median normalised position of the first divergence. Matched denominators, used in Figure 5b, are in §3.

Arm	Engine, cache	Serialised	Domain	cells	bad	ident.	pos
mechanism: engine × cache, matched airline tasks, config C1							
`E4a`	vLLM, **on**	asserted†, unverified	airline	60	0	0/20	0.125
`E12`	vLLM, off	yes, 1 rep.	airline	60	0	20/20	n/a
`E13`	SGLang, **on**	yes, 1 rep.	airline	60	0	0/20	0.048
`E11s`	SGLang, off	yes, 1 rep.	airline	60	0	20/20	n/a
`E11m`	SGLang, off	yes, N rep.	airline	60	0	20/20	n/a
the fix across domains, config C1							
`E4r`	vLLM, **on**	yes†, 1 rep.	retail	60	0	0/20	0.032
`E12r`	vLLM, off	yes, 1 rep.	retail	60	0	20/20	n/a
`E4t2`	vLLM, **on**	yes, 1 rep.	telecom	36	3	1/11	0.355
`E12t`	vLLM, off	yes, 1 rep.	telecom	60	0	20/20	n/a
`E4b2`	vLLM, **on**	yes, 1 rep.	banking	60	0	4/20	0.078
`E12b`	vLLM, off	yes, 1 rep.	banking	60	3	7/19	0.097
serialisation control, and the primary factorial ($R = 5$; B–D add 750 cells each)							
`E4a2`	vLLM, **on**	yes, N rep.	airline	60	0	0/20	0.032
`E1b`	vLLM, **on**	**no**, autoscaled	airline	250	0	0/50	0.032
`E3ret`	vLLM, **on**	**no**, autoscaled	retail	250	0	0/50	0.032

The 19 bad cells are all in C4 retail, and their structure is not uniform. In the cache-off arm, 9 cells are three whole task ids (2, 3, 24) losing all three replicates. In the cache-on arm, 10 cells are three whole task ids (2, 21, 24) plus a single lost replicate of task 4, which is therefore a partially observed task rather than a whole one. Each arm retains 17 tasks with at least one usable cell. Because transcript identity needs at least two usable replicates in *both* arms, the matched C4 retail denominator is 16, while the arm-level counts in this table are over each arm's own 17 usable tasks. The published C4 retail arm in Table 13 carries a denominator of 17 for the same reason, so the losses are a property of that configuration and not of the cache manipulation. Matched denominators in Table 6 are therefore 20, 20, 20 and 16.

Unlike the earlier arms, each of these wrote a manifest recording the resolved model ids and the cache setting. That was added after a manifest naming the wrong agent was found here: the writer had the C1 model ids hardcoded, so a C2 arm recorded `Qwen3-30B-A3B` while all 60 of its cells had used `Qwen3-32B`. The ids in this table are read back from the cells.

Mechanism, generalisation and remedy arms. The clean-substrate factorial and the deterministic-mode pair, plus the residency probes, which are single requests rather than conversations.

Arm	Cfg	Engine, cache	Serialised	Design	cells	bad	ident.
E19	C1	vLLM 0.11.0, off	yes, N rep.	air, A–D, 50×5	1,000	0	A: 50/50
E20a	C1	sglang 0.5.3, **on**, fa3, det off	yes, 1 rep.	air, A, 20×3	60	0	2/20
E21a	C1	sglang 0.5.3, **on**, fa3, det **on**	yes, 1 rep.	air, A, 20×3	60	0	20/20
E26	C1	vLLM 0.11.0, **on**, serialised‡	yes, 1 rep.	air, A, 50×5	250	0	48/50
E22d	C1	vLLM 0.11.0, **on**, FlashAttention	yes, 1 rep.	air, A, 20×3	60	0	0/20
E22t	C1	vLLM 0.11.0, **on**, Triton	yes, 1 rep.	air, A, 20×3	60	0	1/20
E22f	C1	vLLM 0.11.2, **on**, FlashInfer	yes, 1 rep.	air, A, 20×3	60	0	1/20
E23v	C1	vLLM 0.11.2, **on**, default	yes, 1 rep.	air, A, 20×3	60	0	0/20
E23b	C1	vLLM 0.11.2, **on**, batch-inv.	yes, 1 rep.	air, A, 20×3	60	0	9/20
E24t	C1	vLLM 0.11.0, off, Triton	yes, 1 rep.	air, A, 20×3	60	0	20/20
E25r	C1	vLLM 0.11.0, **on**, thinking	yes, 1 rep.	air, A, 20×3	60	0	17/20
E25n	C1	vLLM 0.11.0, off, thinking	yes, 1 rep.	air, A, 20×3	60	0	20/20
E02	C1,C2,C7	vLLM 0.11.0, **on**	yes, 1 rep.	air, A, 50×1	134	0	n/a
ALF on/off	C1 agent	vLLM 0.11.0, on / off	yes, 1 rep.	ALFWorld, 20×3	60+60	0	0/20, 20/20
HIST on/off	C1 agent	vLLM 0.11.0, on / off	yes, 1 rep.	ALFWorld, 5 hist. $\times$ 20	100+100	0	cross-history
probes	C1	vLLM 0.11.0, **on**, 3 configs	yes	7 runs, 5–8 req. each	n/a	0	Fig. 7

‡E26's serialisation is verified from its container launch log. `E19` autoscales across containers while serialising within each; that is safe for a cache-off arm because `E11m` had shown 20/20 under the same topology, and it is what made a 1,000-conversation arm tractable. The `E20a`/`E21a` pair records its engine-parsed server arguments, read live from each server during the arm, alongside the manifest.

Q SERVING MANIPULATIONS

Endpoint routing. The unmodified benchmark at commit `363133a` hardcodes the NL-assertion judge's model (`DEFAULT_LLM_NL_ASSERTIONS`, a hosted model we do not use) in its configuration module and offers no flag for it. Our vendored copy carries a two-file patch, supplied in the supplementary material as a diff against that commit (`tau2_patch_363133a.diff`). In `config.py` the patch makes the judge's model id read `EVALPOWER_JUDGE_MODEL` and its `api_base` read `ENDPOINT_JUDGE`, which the harness sets to the simulator model id and the simulator server's base URL, so the judge is served by whichever model is acting as simulated user, from the simulator container, at temperature 0. The same patch redirects the benchmark's optional model-backed environment-interface class to that endpoint, but the banking domain never instantiates that class: it ran the offline `bm25_grep` retrieval variant, recorded per cell as `retrieval_config`, so no environment call reaches any model. In `data_model/message.py` the patch parses tool-call arguments that arrive as a JSON string into a dict (Appendix V). The agent is served from a separate container with its own cache.

Table 12: Serving manipulations. One row changes per arm; everything else is pinned. *Default* is the *engine's* default, not ours: it is what a reported agent evaluation most likely runs, and what the primary factorial (`E1b`, `E3ret`) used. Every mechanism and domain arm was configured to depart from it on the serialisation row, verified for three arms (Table 11), and departs on the cache row where marked there.

Factor	vLLM	SGLang	Default
prefix cache off	`--no-enable-prefix-caching`	`--disable-radix-cache`	on
request serialisation	`--max-num-seqs 1`	`--max-running-requests 1`	batched
tool parsing	`--tool-call-parser hermes`	n/a	n/a
context length	`--max-model-len` per model	per model	n/a
serving replicas	container count	container count	autoscaled
deterministic kernels	n/a	`--enable-deterministic-inference`	off
attention backend	engine default	`--attention-backend fa3`	engine default
prefill chunking	`--max-num-batched-tokens`	n/a	engine default
KV block size	`--block-size`	n/a	engine default

An observation on the simulator, not a controlled comparison. C4 and C5 differ only in the simulator. Swapping `Qwen3-8B` for `gemma-3-12b-it` moves airline from 1/20 to 4/20 and pushes the median first divergence from 0.147 to 0.250 of the conversation, while swapping the *agent* across C1, C2 and C4 moves those numbers hardly at all. The role of the first divergence shifts too: under C5 it is overwhelmingly the agent (12 of 16 airline tasks), where under C1 it was overwhelmingly the simulator. With the Hermes agent held fixed, changing the simulator changes both the repeatability and the first-discrepancy pattern; these arms do not establish that simulator

Table 13: Generalisation arms, cache on, condition A, 20 tasks $\times$ 3 replicates per domain; entries are tasks reproducing exactly, over tasks with at least two usable replicates. C5 cannot run banking or telecom, because gemma-3 has no tool-calling support and those are the two domains whose simulated user has tools of its own.

Cfg	Agent	Simulator	airline	retail	banking	telecom
C1	Qwen3-30B-A3B (MoE)	Mistral-7B-Instruct-v0.3	0/20	0/20	4/20	1/11
C2	Qwen3-32B (dense)	Mistral-7B-Instruct-v0.3	1/20	1/20	2/20	8/20
C4	Hermes-3-Llama-3.1-8B	Qwen3-8B	1/20	2/17	0/20	7/20
C5	Hermes-3-Llama-3.1-8B	gemma-3-12b-it	4/20	3/17	n/a	n/a

changes have larger effects than agent changes. C5 covers only two domains, because gemma-3 cannot serve tool-calling roles, so this rests on airline and retail alone.

Table 14: Components by replicate count, airline, subsampled in trial order from the same $R = 5$ runs. The $R = 2$ row is one of ten possible replicate pairs and the only one placing $\hat{V}_{\text{agent}}$ below zero; the enumeration is in the text above.

R	$\hat{V}_{\text{infra}}$	$\hat{V}_{\text{sim}}$	$\hat{V}_{\text{agent}}$	CI width
2	0.040	**0.090**	**−0.010**	0.140
3	0.060	0.060	0.013	0.073
4	0.065	0.050	0.023	0.065
5	0.058	0.044	0.040	0.058

Table 15: Between-task share of variance (ICC), airline, $R = 5$. A benchmark exists to separate tasks; by condition D most of the variance no longer does.

Cond.	agent, sim. T	ICC
A	0, 0	0.574
B	0, 1	0.387
C	1, 0	0.318
D	1, 1	0.202

R THE TWO SERVING STACKS, AND HOW THE ENGINE WAS DISENTANGLED FROM THE CACHE

The engine $\times$ cache factorial in Figure 5a did not arrive in that shape. It is worth recording how it got there, because the first version of the comparison was confounded and the confound was the kind that survives review.

What is matched. All five arms use the same 20 airline task ids, the same 3 replicates, the same agent and simulator weights (`Qwen3-30B-A3B` / `Mistral-7B-Instruct-v0.3`), the same temperatures ($T = 0$ on both sides), the same reasoning-mode status, the same 30-step and 1024-token caps, the same zero-retry policy, and requests serialised as the intended setting on one serving replica (`E11m`, which autoscales, excepted; the setting is read back only for the later arms, Appendix P). The two engines differ in implementation, not in what they are asked to do: vLLM's paged prefix cache (Kwon et al., 2023) and SGLang's RadixAttention (Zheng et al., 2024) are independent implementations of the same optimisation, which is why the pattern holding across both is the load-bearing part of the result.

The confound, and how it was resolved. We first ran SGLang against the existing vLLM baseline and observed 20/20 against 0/20. Read alone, that looks like an engine effect, and we initially framed it as one. It was not: the SGLang arm had been launched with its radix cache *disabled* while the vLLM baseline used the engine default, which enables prefix caching. Engine and cache policy were varied together. Two further arms separate them: vLLM with `--no-enable-prefix-caching` (`E12`), and SGLang with its radix cache left *on* (`E13`). Both land where the cache hypothesis predicts and the engine hypothesis does not. The completed 2×2 is what Figure 5a reports; the engine reading is not adopted.

Cost asymmetry, for anyone repeating this. The two stacks were not equally cheap to bring up, and the difference is worth stating because it is a real cost of cross-engine validation rather than a scientific confound. vLLM ran on a slim Debian image without incident. SGLang took six attempts: a `transformers` version conflict pulled in by a quantisation dependency; a hard image-build failure when that version was forced; a CUDA-home assertion in a JIT path that fires

at module import, so disabling the JIT at runtime does not help; and a missing `libnuma` on the CUDA development base image that finally worked. Nothing here affects the comparison, which used the same weights, task ids and decoding, but a reader planning a two-engine replication should budget for it.

S SOFTWARE VERSIONS, AND WHAT WE DID NOT RECORD

The result is configuration-sensitive, so we give the configuration we can recover from the deployment images and say plainly what was not recorded. Notably, both engines ran in a mode where prefix caching is enabled by default, which is what makes the cache-on arms the setting a practitioner would get without choosing it.

Table 16: Serving stack, as pinned in the deployment images. The lower block is configuration that a fully reproducible report should carry and that we did not record per arm; we list it so the gap is explicit rather than implied.

	vLLM arms	SGLang arms
engine version	`vllm==0.11.0`; `0.11.2` for `E22f`, `E23v`, `E23b`	`sglang[all]==0.5.2`; `0.5.3` for `E20a`/`E21a`
engine mode	V1 (`VLLM_USE_V1=1`)	default
base image	Debian slim, Python 3.12	`nvidia/cuda:12.8.1-devel-ubuntu22.04`, Python 3.12
attention backend	engine fallback (FlashInfer not installed)	engine default; `fa3` for `E20a`/`E21a`
accelerator	H100 80 GB agent, L40S 48 GB simulator	H100 80 GB agent, L40S 48 GB simulator; both H100 for `E20a`/`E21a`
replicas	1 unless Table 11 says otherwise	1 unless stated
warm-up	separate logged step before dispatch	separate logged step

Not recorded for the original arms: resolved CUDA, PyTorch, Triton and Transformers versions; dtype and quantisation; cache block size and hashing configuration; model and tokenizer revision hashes; cache-hit rates; whether cache state was flushed between arms; request ordering. *Recorded for the mechanism, generalisation, second-benchmark and remedy arms:* resolved model ids and cache setting per arm (manifests); cache-hit and query counters per request, block size and prefill chunking, and the hashing algorithm (`sha256`) for the probes; engine-parsed server arguments read live for `E20a`/`E21a`.

Endpoints were warmed before dispatch as a separate logged step, so cold-start latency does not contaminate per-call timings. The original arms did not flush cache state between arms or tasks and did not record hit rates, which is why they could show only that the flag controls repeatability. The probes of §4.4 supply what was missing: a fresh container is the flush, the first request is the prime, and the engine's counters log hit and miss per request.

T PRE-REGISTERED AND POST HOC ANALYSES

Table 17 records which analyses were planned before data collection and which were added after the primary results were inspected. Everything from the transcript-identity measure onward is post hoc, and the main text says so where each result is reported.

U ESTIMATOR IMPLEMENTATION AND THE HIERARCHICAL CROSS-CHECK

A pre-registered mixed model with a task random intercept, fitted per condition by REML, reproduces the primary components of Eq. (1) to five decimals. This confirms the estimator was implemented correctly and would have caught mis-weighted pooling, but it is not independent corroboration: the design is balanced at $R = 5$, and for a balanced one-way random-effects model the REML residual equals the pooled within-task variance analytically. We report it because the pre-registration promised it, with that limitation stated rather than dressed up as support. The genuinely new quantity from that model is the ICC in Table 15.

Table 17: Which parts of this paper were planned in advance. The mechanism story is post hoc: the temperature factorial was the pre-registered study, and the cache result came out of its control condition.

Analysis	Status
Temperature factorial, reward variance	pre-registered
Variance-component estimator and bootstrap	pre-registered
Hierarchical (REML) cross-check	pre-registered
Transcript identity	added after inspecting the primary results
Engine × cache factorial	post hoc mechanism experiment
Cross-domain cache-off arms	post hoc generalisation and boundary
Replica and serialisation controls	post hoc diagnostic controls
Replicate-pair enumeration	post hoc
Exact intervals and paired tests on every count	post hoc
Three-architecture cache-off generalisation (§4.4)	post hoc
Cache-off factorial (Appendix D)	post hoc
Throughput and deterministic-mode arms (§4.4)	post hoc
Residency probe and its two controls (§4.4)	post hoc
Attention-backend sweep and batch-invariant arm (§4.4)	post hoc
Second benchmark (§4.4)	post hoc
Decision-relevance re-analysis (§4.4)	post hoc
Serving-configuration sensitivity (§4.4)	post hoc
Reasoning-enabled scope arm (App. N)	post hoc

V FAILURE ACCOUNTING AND HARNESS DEFECTS

Across the whole programme 62 arms dispatched 7,065 conversations, of which 697 (9.9%) produced no usable reward or transcript; the figure of roughly 1,010 of 4,281 quoted in an earlier version was a snapshot taken midway. The losses concentrate in the pilot series and the early probes (Table 18), four arms lost every cell, and the arms that carry this paper's claims lost 25 cells: 22 across the 33 arms outside the pilot and early-probe groups, and 3 in the early telecom probe `E4t2` that also appears in Figure 5b.

Table 18: Dispatch and loss ledger, from the per-cell records.

group	arms	dispatched	lost	lost %
pilot series E5 to E8 (no serving record; excluded)	17	996	281	28.2
early probes E4*, E2tel, E4bank	9	865	394	45.5
temperature factorials E1b, E3ret, E19	3	3,000	0	0.0
all other reported arms	33	2,204	22	1.0
total	62	7,065	697	9.9

The four causes below account for the concentrated losses. Each is worth stating because each is a way for a configuration to report success while doing something else, the same shape as this paper's main finding.

A Mistral agent configuration lost 240 cells to a tool-call schema mismatch: the engine's parser returns tool-call arguments as a JSON string while the harness requires a dict, so conversations terminate at *zero* messages with a *null* error field, making the cells look empty rather than broken. A gemma-3 simulator lost 120 cells in the two domains needing simulator-side or environment-side tool calls, because that model has no tool-calling support and our parser selection fell back to a default instead of failing loudly. Banking lost 251 cells to an optional retrieval dependency that the benchmark imports but does not declare, so every cell died *after* loading a 698-document knowledge base and looked like a data problem. A task-id filename collision mapped 20 telecom tasks onto 12 files, which is why the telecom cache-on arm covers 11 usable tasks and its comparison in Figure 5b is restricted accordingly.

Total measured compute is 102.7 conversation-hours across all arms, including failed ones. Billed cost at the last reconciliation, taken before the pairing, clean-substrate, remedy and second-benchmark arms, was $157.22 for the project's own workloads; the arms collected after that point are not reconciled here and their cost is larger than that figure.

The 1,600 conversations of the generalisation, clean-substrate and remedy arms lost 19 cells, 1.2%, all in C4 retail: six task-arm losses of whole task ids across four distinct ids, {2, 3, 24} in the cache-off arm and {2, 21, 24} in the cache-on arm, plus one lost replicate of task 4 in the cache-on arm. Tasks losing every replicate are excluded; task 4 enters with its two usable replicates, giving a matched denominator of 16. These are whole task ids losing every replicate to an infrastructure error, plus one task losing a single replicate (Appendix P). Their billing is not yet reconciled. One defect was found and fixed during that collection: the manifest writer had the C1 model ids hardcoded, so a C2 arm's manifest named an agent its cells had not used, and the cache setting was not recorded at all. Both are now read from the run rather than asserted, which is the same lesson as the paper's main finding applied to its own tooling.

W LICENSES FOR EXISTING ASSETS

We use τ^2-bench under the MIT License. vLLM and SGLang are distributed under the Apache License 2.0. The Qwen3 checkpoints used here (`Qwen3-30B-A3B`, `Qwen3-32B`, `Qwen3-8B`) and the Mistral checkpoints (`mistralai/Mistral-7B-Instruct-v0.3`, `mistralai/Mistral-Small-24B-Instruct-2501`) are distributed under the Apache License 2.0. `NousResearch/Hermes-3-Llama-3.1-8B` is used under the Meta Llama community license declared on its model card. The Gemma-3 weights used through `unsloth/gemma-3-12b-it` are subject to the Gemma Terms of Use, while `google/gemma-4-E4B-it`, used only for the sampling-validation run in §3, is distributed under the Apache License 2.0. Checkpoint cards: Mistral AI (2024; 2025); Nous Research (2024); Google DeepMind (2025; 2026); the Qwen3 checkpoints are described in Yang et al. (2025). All assets were used in accordance with their respective licenses and terms.