CANDIDATE ORDER IN LLM-GENERATED AGENT ENVIRONMENTS: PROMPTED PLACEMENT, MEASURED DIFFICULTY AND LEARNED RELIANCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Language models increasingly write the environments used to evaluate and train tool-using agents. Such environments are checked for solvability and difficulty, not for where the correct option sits among an agent's candidates. We study this in a record-chain lookup task in which each record lists one labelled active candidate and two decoys. In fresh generated worlds, Sonnet 5 listed the active candidate first in 85.0% of these choices and two GPT-5.6 generators in all of them; the format example in the generation prompt also listed it first. Three separate controlled studies examine this regularity. Moving the active object to second or third place in the format example reduced first-position placement for all three generators (Holm-adjusted $p \leq 3 \times 10^{-5}$ each), almost completely for one and partially for the other two. On identical worlds, the generators' own order raised traversal completion by 6.4 to 15.2 percentage points for Qwen3-1.7B and by 3.4 to 12.4 points for Qwen3-4B, while re-permuting an already shuffled reference produced no detectable change; the 4B solver's final answers showed no corresponding advantage. In matched fine-tuning of Qwen3-1.7B, permuting the candidate lists inside otherwise identical training trajectories raised held-out traversal completion by 58.6 to 86.8 points across three seed pairs and final-answer success by $+10.46$ points on average; models trained on the generators' order mostly followed a decoy once the order changed. The studies share one task mechanism and one model family and do not test the full prompt-to-learner chain. Generated environments should randomise and report candidate order.

1 INTRODUCTION

Environments for tool-using agents are increasingly written by language models. A generator is asked for tasks, records, tools and a verifier, and its output is admitted if it can be solved and, often, if solver success matches a requested difficulty (Wolf et al., 2026; Wu et al., 2026; Liu et al., 2026; Meng et al., 2026; Guo et al., 2025; Fan et al., 2026). These checks concern how hard an environment is. They do not ask whether its presentation gives the solver a shortcut.

The order of the options an agent must choose among is part of that presentation. Language models are sensitive to position when they read lists (Liu et al., 2024; Tamba, 2026; Blankenstein et al., 2025), and they leave position regularities in what they write: multiple-choice questions written by language models favour model-specific answer slots (Tang et al., 2026). In a generated environment such a regularity affects evaluation and training at once, because the order the generator writes is the order in which solvers are evaluated and demonstrations are rendered.

We study this in a record-chain lookup task (Figure 1a). Each record points to its successor through three candidates, each carrying a status label: one *active* candidate and two *superseded* decoys. The label alone identifies the active candidate, so the task does not require any particular order. Yet in the 12,256 fresh generated worlds analysed here, Sonnet 5 listed the active candidate first in 85.0% of 44,717 branching decisions (records that offer a choice; a per-world mean of 0.738, rising with depth), and Terra and Luna, two GPT-5.6 models, did so in all 23,110 and 25,034 of theirs (Appendix B). A solver that always takes the first candidate therefore never meets a decoy in Terra's

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.

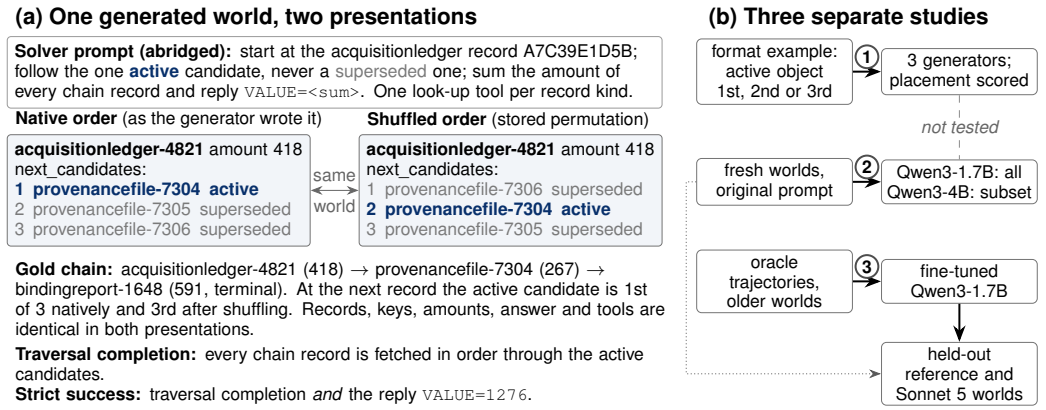


Figure 1: The task and the three studies. (a) A real admitted world written by Terra (`gen-library-d3-i101`), as served to solvers; only the solver prompt is abridged. Each non-terminal record lists one active candidate and two superseded decoys that resolve to real terminal records, and the shuffled presentation applies a stored permutation to each list of the same world. (b) The three studies, each with its own manipulation (numbered arrows): (1) the position of the active object in the prompt's format example; (2) the order in which identical worlds are presented to two solvers; (3) the candidate order inside otherwise identical training trajectories, with the same order of training examples within each seed pair. Worlds from the prompt study were scored for placement only; none was solved or used for training (dashed link). The 4B solver saw a subset of the 1.7B worlds; the training study was evaluated on the reference grid that is also the measurement control and, secondarily, on the measurement study's fresh Sonnet 5 worlds (dotted arrow).

or Luna's worlds, and in Sonnet 5's mainly at shallow depths. The format example in the prompt that produced these worlds also lists the active object first.

We ask three questions and answer each with a separate controlled study that has its own error control (Table 1). Can the format example set where generators place the active candidate? On identical worlds, does that placement change measured solver performance? Can a model trained on the placement learn to rely on it? No study manipulated the prompt and then measured solvers or learners, so together they do not test an end-to-end causal chain (Figure 1b). Our contributions are:

- **Prompted placement (Section 4).** Moving the active object from first to second or third place in the format example lowered first-position placement for all three generators. How closely they reproduced the demonstrated position differed: Terra almost never kept the active candidate first, Luna reproduced a third-place example but largely ignored a second-place one, and under a third-place example Sonnet 5 most often put it second. Admission rates barely changed.
- **Measured difficulty (Section 5).** On identical worlds, native order raised traversal completion for two Qwen3 solvers, while re-permuting an already shuffled reference showed no detectable change. For the 4B solver the advantage appeared in traversal but not in correct final answers. A pre-registered depth test detected depth dependence only for Sonnet 5 worlds.
- **Learned reliance (Section 6).** Fine-tuned on trajectories whose candidate lists always put the active candidate first, Qwen3-1.7B followed decoys once the order changed. Training on the same trajectories with every candidate list permuted, in the same example order, avoided this loss and also raised final-answer success.

On an external benchmark whose stored order is already random, re-permuting the returned records produced no detectable overall change, although one solver's horizon-shape test rejected narrowly (Section 7). Every number in the paper is generated from a hash-checked analysis output (Appendix G).

2 Related work

Generated environments and their validation. Language models now generate tool-use environments, tasks and verifiers for training and evaluation (Wolf et al., 2026; Wu et al., 2026; Liu et al., 2026; Guo et al., 2025; Fan et al., 2026), and procedurally generated environments support

downstream tool-use training (Sullivan et al., 2025). Validation typically targets solvability and difficulty calibration (Meng et al., 2026; Krsteski & Meyer, 2026; Shi et al., 2026). Protocol audits show that benchmarks can reward behaviour other than the intended capability (Shao et al., 2026; Zhu et al., 2026).

Position effects and learned shortcuts. Models that read lists favour some positions: in long contexts (Liu et al., 2024), in multiple-choice answers (Tamba, 2026), where option labels also attract a preference (Zheng et al., 2024), in tool selection (Blankenstein et al., 2025), and increasingly with longer reasoning (Wang, 2026). Few-shot examples whose answers share a position pull a model's answers towards it (Turpin et al., 2023), and skews in the in-context examples of a data-generation prompt carry over into generated tabular data (Recasens et al., 2025). Training data in which answers sit in one position teach models to look there (Ko et al., 2020), tool-use reinforcement learning can learn spurious cues (Yang et al., 2026b), and randomised field order is used to make retrieval encoders order-invariant (Solatorio et al., 2026).

Closest work. Three studies each establish one of the phenomena we build on. Tang et al. (2026) measure where ten language models put the correct answer in multiple-choice questions they write: the distribution over slots is strongly non-uniform, and an instruction to balance it makes it more uniform. Their items are static, their prompts contain no worked example with a fixed answer position, and they do not measure consequences for test-takers or training. Shao et al. (2026) audit agent benchmarks for protocol validity. In a probe they constructed on a causal-discovery task whose data generator emitted variables in topological order, an agent that exploited this order scored far higher without recovering the intended structure, and they recommend shuffling variable order before presenting data. Maniyar et al. (2026) train with reinforcement learning on correctly labelled multiple-choice questions whose answer is always in the first slot and, against the same questions with shuffled positions, find that many models, smaller ones in particular, learn to answer by position.

What the present studies add. We do not claim that generator position bias, exploitable generator regularities or learned positional shortcuts are new. We add controlled evidence of each in generated interactive environments, where the order is a by-product of generation rather than a condition set by the experimenter: a manipulation of one prompt element, the position of the active object in a format example, with everything else fixed; paired presentations of identical multi-step worlds, with a control and with traversal separated from final answers; and a training comparison that changes only candidate order inside otherwise identical tool-use trajectories whose order came from the generators. For environment builders, this separates a presentation artefact from difficulty, gives its size on the metrics they report, and identifies two choices they control: the examples in their generation prompts and the order in which candidate lists are rendered.

3 TASK, GENERATORS AND THE ORDER INTERVENTION

Task. A world is a chain of records, requested at depth $d \in \{1, \ldots, 16\}$, in one of eight domains that share one record-chain mechanism, so they are not eight independent task types. The agent is told the entry record and must follow the active candidates to the terminal record using sixteen look-up tools, one per record kind, then reply `VALUE=<sum of amounts>` (Figure 1a). *Traversal completion*, the primary outcome, records whether the agent's look-ups reach the terminal record along the active chain: each chain record must be fetched in order, by a key the agent could know from the record before it. Look-ups of decoys or of records already seen do not reset progress, so an agent can recover from a wrong look-up within its turn budget and a cap of four look-ups per chain record plus eight. Traversal is graded from the look-up sequence and ignores the final answer. *Strict success* also requires the correct sum. We report the two separately; neither is end-to-end task success in a broader sense. The *verified depth* of a world is the audited number of dependent look-ups on its gold chain, which for generated worlds can differ from the requested depth.

Generators and references. Generated worlds come from `claude-sonnet-5` (Sonnet 5), called through the Claude Code command-line interface at low effort with one isolated invocation per request, and from `gpt-5.6-terra` and `gpt-5.6-luna` (Terra, Luna), called through the chat-completions API at temperature 0.8 with reasoning effort set to none. The prompt fixes the

Table 1: The controlled studies. Each primary analysis was fixed before its outcomes and has its own error control; no study-wide correction is claimed. The 4B panel uses a subset of the 1.7B worlds; the training study is evaluated on the 1.7B study's reference grid and, secondarily, its fresh Sonnet 5 worlds (Figure 1b).

Study	What is varied	Independent unit and sample	Outcome and primary comparison
Placement (Sec. 4)	position of the active object in the prompt's format example; 4 conditions	generation request; 512 per generator in 128 blocks	joint validity-and-first-position score; P1 minus mean of P2 and P3; blocked randomisation test, Holm over 3 generators
Measurement, 1.7B (Sec. 5)	native order vs. a stored permutation of the same world	world; 12,256 admitted generated and 3,072 reference worlds, each in both presentations[a]	traversal completion; primary: two-sided conditional depth test, Holm over 3 generators; paired gains descriptive, world-cluster bootstrap
Measurement, 4B (Sec. 5)	the same presentations	world; 1,024 per pool, a subset of the 1.7B worlds	traversal completion (strict answer separate); exact McNemar test, Holm over 3 generators; control outside the family
Training (Sec. 6)	candidate order inside otherwise identical oracle trajectories (NATIVE vs. RANDOM); same example order within a seed pair	fitted model; 3 seed pairs; 3,072 held-out reference worlds	traversal on REF; RANDOM minus NATIVE, mean of seed pairs; interval conditional on the fits

[a] These are the native/shuffled comparisons. A separate exploratory arm serves the reference worlds once with rewritten instructions (3,069 analysed presentations), for 33,725 analysed of 33,792 scheduled solver rows across all arms; the other 67 rows belong to inadmissible generated worlds or rewrites.

schema, two decoys per step and the requirement that every candidate be a real record (Appendix A). An output is admitted if it materialises into a chain and an oracle replay of the gold chain returns the correct sum. Inadmissible outputs are kept as outcomes in the generation counts and never replaced; solver comparisons use admitted worlds. The programmatic reference REF has exact depth and candidate lists that are already shuffled per item.

The order intervention. The *native* presentation is the generator's own order. The *shuffled* presentation applies a stored, uniformly drawn permutation to each candidate list of the same world, chosen without reference to which candidate is active. Records, keys, amounts, gold chain, answer, depth, tools and instructions are identical, and this invariance was checked for every admitted world. REFS applies the same rule to REF; since REF is already shuffled, no effect is expected, so REF versus REFS is a control.

4 THE FORMAT EXAMPLE INFLUENCES PLACEMENT

Design. The generation prompt contains a JSON skeleton whose `next_candidates` field shows the expected format. In the original prompt (O), used for all fresh worlds above, that field is schematic: an active object is written first, followed by the text "EXACTLY 2 with status superseded". Three further conditions replace it with three fully written example objects, the active one listed first (P1), second (P2) or third (P3). Nothing else differs: the rule text, record-existence requirement, counts, domain, depth and variation clause are identical, and no condition asks the generator to shuffle or to copy the example's order. Each generator received 128 blocks (eight domains, requested depths 2, 6, 11 and 16, four replicates), each holding one request per condition, for 1,536 requests in all, through the same routes and settings as the fresh worlds.

Estimand. The pre-registered per-world score is the fraction of the world's branching gold-chain decisions whose first listed candidate is active, if the world is admitted and has at least one such decision, and 0 otherwise. It is therefore a joint measure of validity and first-position placement. The primary contrast T is the mean score under P1 minus the mean of P2 and P3, taken within blocks: it measures how far first-position placement falls when the example moves the active object out of first place. It does not measure whether a generator reproduces the demonstrated position,

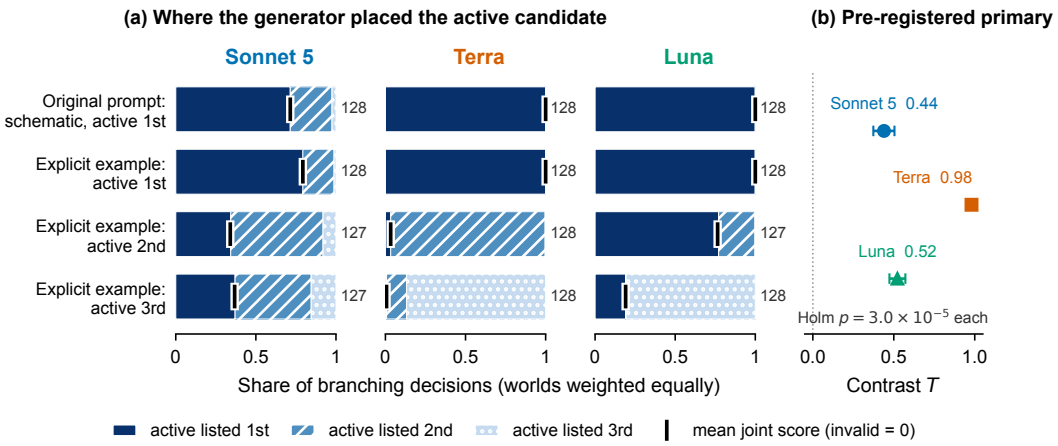


Figure 2: Moving the active object in the format example changes where generators place it. (a) For each generator and prompt condition, the share of branching gold-chain decisions with the active candidate listed first, second or third, averaging per-world shares over admitted worlds; the number of admitted worlds (of 128 requests) is printed at the right. The black tick is the mean pre-registered joint score, which also counts inadmissible outputs as 0. (b) The primary contrast T, the fall in the first-position score from P1 to P2 and P3, with 95% block-bootstrap intervals over 128 blocks; Holm-adjusted $p = 3.0 \times 10^{-5}$ for each generator, the smallest value the randomisation test can produce after adjustment.

which we describe separately. T is tested by a two-sided blocked randomisation test (100,000 draws, so the smallest attainable p is about 10^{-5}) with Holm adjustment over the three generators; intervals come from a block bootstrap.

First-position placement falls for all three generators. The primary contrast is $T = 0.980$ for Terra, 0.522 for Luna and 0.439 for Sonnet 5, and all three reject after Holm adjustment (Figure 2b). Where the active candidate goes instead differs by generator; the shares below are per-world shares of branching decisions averaged over admitted worlds, the pre-registered convention. Terra follows the example closely: with the active object third, its worlds place the active candidate third in 86.8% of their decisions on average. Luna reproduces a third-place example (80.6% third) but largely ignores a second-place one (77.1% still first). Sonnet 5 moves the active candidate out of first place only partly and does not copy the demonstrated position: under P3 its worlds list it first in 37.0% of decisions, second in 47.5% and third in 15.5%. Differences between generators are confounded with their generation interfaces and settings.

Validity does not explain the effect, but it is part of the endpoint. Sonnet 5 admits 128, 128, 127 and 127 worlds under O, P1, P2 and P3; its two failures are malformed JSON, kept as scored outcomes. Its admission contrast is 0.008 [0.000, 0.020], and the placement effect among admitted worlds alone, 0.432 [0.367, 0.498], is close to the joint estimate. Depth fidelity, which the score does not include, varies more, mainly for Luna (Table 2). One Sonnet 5 response is a recorded recovery after a first attempt with an unknown provider outcome; it barely affects T (Appendix A).

The original schematic example differs from an explicit one (secondary). For Sonnet 5 the schematic original prompt gives a lower joint score than the explicit first-place example: O minus P1 is -0.079 [-0.150, -0.007], raw $p = 0.035$, a secondary comparison without multiplicity control. Terra and Luna are at ceiling under both. The experiment shows that the position of the example is sufficient to change placement in all three generators. It does not show that the example was the only cause of the ordering in the fresh worlds, or that the generators have an intrinsic positional preference.

5 Candidate order changes measured traversal

Solvers and presentations. The main measurement served every admitted fresh world in both presentations to `Qwen3-1.7B` (thinking off, temperature 0, one rollout per presentation, a turn

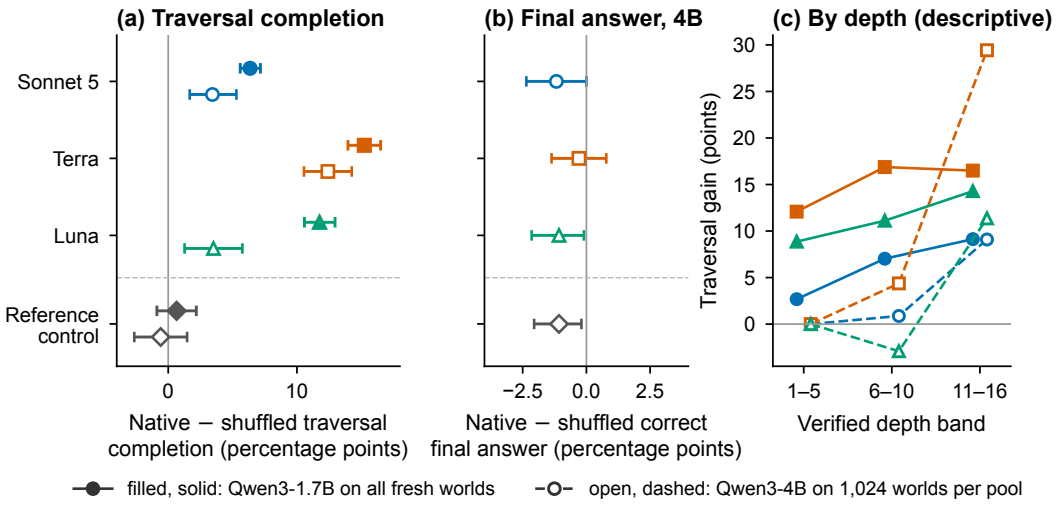


Figure 3: Native order raises traversal completion; the 4B solver's final answers show no such advantage. (a) Native minus shuffled traversal completion on matched worlds, with 95% stratified world-bootstrap intervals, for Qwen3-1.7B on all admitted fresh worlds (6,113, 3,072 and 3,071 generated; 3,072 reference) and Qwen3-4B on the 1,019, 1,024 and 1,024 matched worlds of its subsample (1,024 reference). The reference control re-permutes already shuffled lists. (b) The same contrast for correct final answers (strict success), Qwen3-4B, same denominators. (c) Traversal gains by verified-depth band, without intervals; they are descriptive and belong to no test family. Panels (a) and (b) have different horizontal scales. The rows are separate analyses: the 1.7B intervals are descriptive, while the 4B generator rows form one Holm family; all intervals are pointwise.

budget of $2\max(d_{\text{requested}}, d_{\text{verified}}) + 3$). This solver was the only one of ten screened candidates that met a pre-registered rule for a usable measuring instrument: competent at depth 1, no sweeping of the key space, and a usable range below ceiling. The fresh pools hold 6,144 Sonnet 5 requests and 3,072 each for Terra and Luna, plus REF and REFS grids of 3,072 reference worlds (the training study's held-out grid) (Table 1 and Appendix C give the analysed populations). A confirmation panel used a second, more capable solver from the same family, `Qwen3-4B-Instruct-2507`, which a later re-screen at greater depths had found competent and below ceiling (traversal range 0.75 across the screened depths, 0.25 at depth 16); it was chosen on those criteria, never on an order effect. It was served with its own chat template (it has no thinking mode), temperature 0 and the same budgets, on a fixed subsample of the same worlds (eight per domain and requested depth, 1,024 per pool) with the original stored permutations. Of 8,192 scheduled presentations, 8,182 were analysed; the other 10 belong to 5 Sonnet 5 worlds that were already invalid.

Estimands and uncertainty. The paired gain is native minus shuffled traversal completion on matched worlds, in percentage points, with 95% intervals from a stratified world-cluster bootstrap that keeps both presentations of a world together; intervals here and below are pointwise and not adjusted for multiplicity. For the 1.7B solver the pre-registered primary is a test of depth dependence, described below, and the paired gains are descriptive. For the 4B solver the primary is an exact two-sided McNemar test on discordant pairs with Holm adjustment over the three generator pools; the reference control lies outside that family, and strict success is a separate secondary endpoint.

Native order raises traversal for both solvers; the control stays near zero. For Qwen3-1.7B, native order raises traversal completion by $+6.36$, $+15.23$ and $+11.76$ points for Sonnet 5, Terra and Luna, while re-permuting the reference changes it by $+0.65$ (Figure 3a). For Qwen3-4B the gains are $+3.43$, $+12.40$ and $+3.52$ points, with Holm-adjusted $p = 0.002$, 1.4×10^{-27} and 0.007; the control gives -0.59, $p = 0.675$. The control estimates are small and their intervals include zero, but no equivalence margin was fixed in advance, so they do not show that permutation is neutral. Because both presentations are the same worlds, the gains are effects of order, not of content. On identical worlds the 4B gains are smaller than the 1.7B gains for Terra and Luna (Appendix C).

For the 4B solver the advantage is in traversal, not in final answers. For Qwen3-4B, native minus shuffled strict success is -1.18, -0.29 and -1.07 points (Holm $p = 0.24$, 0.74 and 0.24; Figure 3b). The traces show why. Across all eight pool and presentation combinations of the panel, 6,630 traversals were completed but only 1,959 ended with the correct sum; the other 4,671 reported a parseable but wrong value. Summing the amounts is where this solver most often fails, and the extra traversals it completes under native order do not become more correct answers. If anything, strict success is slightly lower under native order, including for the reference control (-1.07, raw $p = 0.052$). The 1.7B solver rarely answers correctly in either presentation, for example in 225 and 209 of 6,113 Sonnet 5 worlds (Appendix C). With these solvers, a builder who validated generated environments by final-answer success alone would not see the order effect that traversal shows.

The advantage appears where the solver is below ceiling. The 4B solver completes every world of verified depth 1 to 5 in both presentations, so its gains there are zero; they concentrate in depths 11 to 16, where they reach $+9.1$, $+29.4$ and $+11.4$ points. The 1.7B solver shows gains in every band (Figure 3c). For this solver the pre-registered primary asked whether, among discordant worlds (one presentation succeeds and the other fails), the odds that the native presentation wins depend on verified depth. The test is two-sided, conditions on each domain's number of native wins and allows a constant domain-specific preference (Appendix C). It rejects for Sonnet 5 (Holm $p = 1.8 \times 10^{-4}$) but not for Terra (0.950) or Luna (0.169), the same pattern as in a corrected repeat on older worlds from the same generators. Sonnet 5's advantage does not change monotonically with depth (Appendix C). Sonnet 5 is also the only generator whose placement varies with depth, from 3.1% active-first decisions at requested depth 2 to 90.9% at depth 16 (Appendix B); this may contribute to its depth-dependent advantage, but it is a post hoc observation, not a tested explanation. The Terra and Luna nulls are not evidence of flat effects: a secondary within-domain statistic rejects for both in its own Holm family (0.019 and 0.019), and the conditional test has little information for them, since Luna's 413 discordant worlds include only 26 shuffled wins and Terra's 516 only 24.

Sensitivities and a wording comparison. Excluding the worlds whose advertised decoys do not all resolve, identified per generator pool, leaves the 4B gains at $+3.43$, $+12.33$ and $+2.57$ points. Luna's raw exact p in this sensitivity is 0.0504, so it rejects for Sonnet 5 and Terra but not for Luna, whose 4B gain is the least robust. For the 1.7B solver the pre-declared depth-test sensitivities change no decision, although Luna's comes within Monte Carlo error of the threshold on requested depth (Appendix C). An exploratory comparison with rewritten instructions does not isolate an effect of wording: the reference served with instructions rewritten by Luna completed 0.713 of its worlds against 0.861 with the original wording, but the comparison was made after outcomes and in a separate solver session, and 679 of the rewrites are ambiguous about which amounts the sum covers (Appendix C). The generated worlds share the original instructions, so the paired order contrast does not depend on this comparison.

6 Training on active-first trajectories teaches reliance on position

Intervention. We fine-tuned the measurement solver, Qwen3-1.7B, on verified oracle trajectories for 3,072 older Luna and Terra worlds, with 512 further worlds for development; no Sonnet 5 world and no test world was used. Each trajectory was rendered twice, differing only in the order of the candidate lists inside its tool responses. In NATIVE order, as the generators wrote them, the active candidate was first in 23,992 of 23,992 gold-path decisions. In RANDOM order, one stored permutation per list put it first, second or third in 7,991, 7,984 and 8,017 decisions. Targets, loss masks, record content, token counts and the order of training examples are matched within each pair.

Training. Six full-parameter fits, NATIVE and RANDOM at paired seeds 17, 29 and 43, each ran 576 AdamW updates over 3 epochs at learning rate 1×10^{-5} (Appendix D). This is supervised fine-tuning on oracle labels, not reinforcement learning. A prefix-preserving chat template was adopted for training and serving before any evaluation; the untrained BASE model was served with the same template, so it is a reference for this experiment rather than a general capability score, and its rate is not comparable with that of the same model in Section 5.

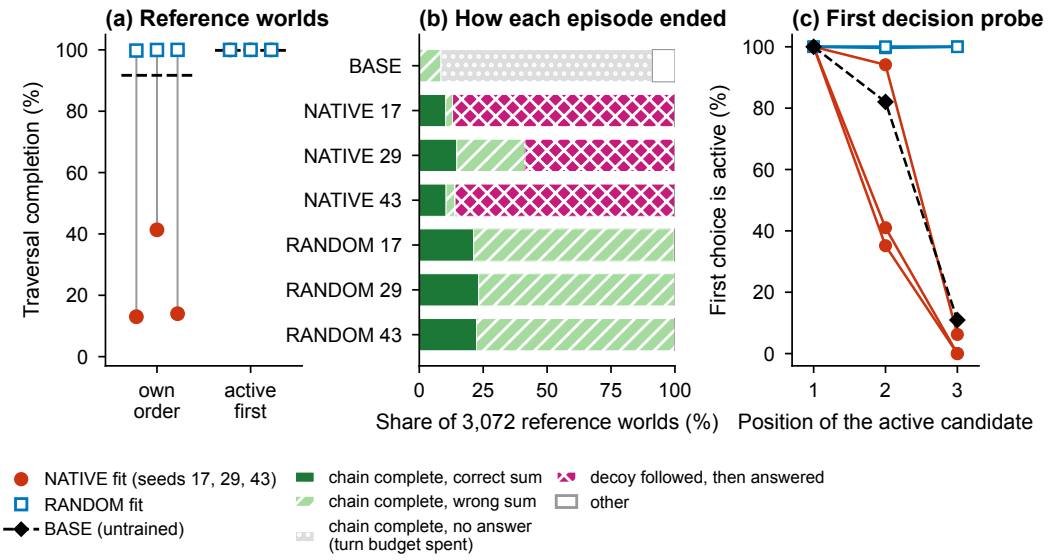


Figure 4: Matched fine-tuning. NATIVE and RANDOM fits were trained on the same trajectories, with candidate lists in the generators' order or permuted. (a) Traversal completion (equal cell weights) of each fit on the 3,072 reference worlds in their own order and with the active candidate first; lines join the members of a seed pair, and the dashed line is BASE. (b) How each of the 21,504 episodes on the reference worlds in their own order ended, from a replay of the stored traces (categories confirmed by an independent replay). (c) Teacher-forced first decision on 256 reference worlds with the active candidate placed first, second or third: the share of fits' first choices that name it. All seeds are shown; RANDOM fits overlap at the top.

Evaluation and uncertainty. The seven checkpoints were evaluated on the held-out reference grid, the same worlds as the measurement control, in its own order (REF) and with the active candidate first (REFA), and, as a secondary evaluation, on the 3,055 valid worlds of a reserved subset of the measurement study's fresh Sonnet 5 worlds in both presentations: 86,016 scheduled episodes across the four arms. The primary estimand, fixed before evaluation, is RANDOM minus NATIVE traversal on REF, weighting the 128 domain-by-depth cells equally and averaging the three seed pairs. Its interval covers held-out worlds conditional on the six fits, not retraining variability; the three seed pairs are the only replication of training.

Permuted candidate lists prevent a large loss. Per seed pair, RANDOM minus NATIVE REF traversal is $+86.75$, $+58.63$ and $+86.00$ points; the mean is $+77.13$ [76.35, 77.93], while the seed standard deviation of 16.03 points is not reflected in the narrow conditional interval (Figure 4a). NATIVE fits average 22.8%, 68.91 points below BASE at 91.7%, and RANDOM fits are $+8.21$ points above it. BASE is not a clean reference, however: it keeps looking up records after the terminal one and so can recover from a wrong look-up, whereas the fits answer once they believe the chain is complete. Comparisons with BASE therefore mix reliance on position with stopping behaviour; RANDOM minus NATIVE, whose fits share training data and stopping behaviour, is the clean contrast.

The loss is presentational. Listing the active candidate first restores the NATIVE fits (REFA gains of $+86.9$, $+58.7$ and $+86.0$ points), and RANDOM minus NATIVE on REFA is $+0.02$ [0.00, 0.07]. In the stored traces most NATIVE-fit episodes end after the model follows a decoy and then answers: 2,671, 1,802 and 2,641 of 3,072 (Figure 4b). In a teacher-forced probe of the first decision, the NATIVE fits choose the active candidate in 0, 16 and 0 of 256 cases when it is listed third, and the RANDOM fits in at least 255 of 256 at every position (Figure 4c; Table 9). With the active candidate third, the NATIVE fits often pick the *second*-listed decoy, so the learned reliance is not a literal rule to take the first candidate. Seed 29 relies on position much less than seeds 17 and 43, which is why the seed spread is large.

Final answers improve, but much less than traversal. Unlike the 4B comparison of Section 5, which varies the presentation for a fixed model, this comparison varies the training data, and it does change final answers: RANDOM minus NATIVE strict success on REF is $+10.46$ points [9.78, 11.15], against $+77.13$ for traversal. RANDOM fits complete nearly every chain but more often report a wrong sum than a correct one (Figure 4b). BASE never answers correctly on REF: it completes 2,817 chains, but in 2,540 episodes it keeps calling tools after the terminal record until its turn budget runs out, so its zero final-answer rate reflects a failure to stop, not its navigation.

Held-out generator (secondary). On the fresh Sonnet 5 worlds, RANDOM minus NATIVE traversal is $+16.03$ [15.13, 16.97] points in native order and $+62.71$ [61.63, 63.78] after shuffling, with BASE at 96.9% and 94.5%. The contrast is smaller in native order, which is consistent with Sonnet 5 itself listing the active candidate first in most decisions. These worlds come from the measurement study, so this is a transfer test to a held-out generator, not an independent replication.

7 A SCOPE CHECK ON AN EXTERNAL BENCHMARK

To see whether display order matters in an independently built workflow, we used AgentCE-Bench (Yang et al., 2026a), a constrained grid-planning benchmark, with 512 fresh worlds, two solvers (`Qwen3.5-9B` and `Granite-4.2-8B`) and horizons 1, 5, 11 and 21 (Appendix E). Its stored record order is already random, so there is no natural first-is-correct baseline; the shuffled presentation re-permutes only the returned records. Neither solver shows a detectable overall change: shuffled minus stored completion is $+0.80$ [-3.19, 4.79] points for Qwen3.5-9B and -0.83 [-2.80, 1.32] for Granite, intervals that do not establish equivalence. Granite's primary horizon-shape test rejects narrowly (Holm $p = 0.0474$) and Qwen's does not (0.4582). This measures sensitivity to re-permuting already random records; it tests neither the native-order effect nor the transfer of the trained policies.

8 DISCUSSION

What the studies show together. In this pipeline, one element of the generation prompt can influence candidate order, that order accounts for a measurable part of the traversal difficulty two solvers experience, and a learner trained only on the generators' order comes to rely on it. Each statement rests on its own intervention. The prompt-study worlds were neither solved nor used for training, so how much of the measurement or training effect would disappear under a different prompt is an inference, not a measured quantity.

Implications for environment builders. Validation by solvability, or by solve rates measured on a single presentation, cannot separate this regularity from difficulty. What a builder sees also depends on the metric. For the 4B solver, native order raised traversal completion without a detectable final-answer advantage, so validation by final answers alone would miss it; in training, by contrast, permuting candidate lists improved final answers as well as traversal. The remedies are cheap: randomise every unordered candidate list per item and record the permutation; report the active-first frequency of a generated pool; validate difficulty with paired presentations of the same worlds; and treat format examples in generation prompts as design choices that can set structural properties of the output.

Limitations. The three main studies use one record-chain mechanism across eight domains, and their solvers and learner belong to the Qwen3 family; stronger screened solvers were at ceiling on this task, so the solver-side results are not evidence of broad generality. The training study fine-tuned one learner with three seed pairs, and its NATIVE data always listed the active candidate first; its intervals are conditional on the fits, and the effect of partially ordered data is untested. No study tested the path from prompt to learner. We also did not establish how order compares with other presentation factors, because the wording comparison in Section 5 is not a clean estimate of wording sensitivity. Sonnet 5's fresh worlds came through a command-line interface rather than the direct API used for older worlds, several analyses share worlds (Figure 1b), and the corrected repeat re-uses older worlds. Appendix F records the chronology, the amendments, and the inference defects of an earlier exploratory study, which were corrected before the studies above.

AI USE STATEMENT

The authors directed the study and take responsibility for its content. Generative AI assisted with research formulation and design (conceptual framing, hypotheses, mathematical claims and experiments); method implementation and synthetic-data generation and preparation; and qualitative analysis, interpretation, literature review, manuscript drafting and figures. AI was not used for translation; proof assistance was not applicable.

ETHICS STATEMENT

The environments are synthetic record systems with random keys and amounts; no personal data and no human subjects are involved. Six models were fine-tuned from an open-weight 1.7B model on synthetic oracle trajectories. The programme's recorded consumption, as list-price estimates, was $241.27 (GPU $186.76, API $47.43, CPU estimates $7.09; Appendix F). The main risk is misreading a scoped result as general; our findings are specific to the task, solvers, generators and interventions studied.

REPRODUCIBILITY STATEMENT

Appendix G describes the anonymous supplementary archive, which one command runs. It regenerates every macro, table and figure-data file of this paper from hash-checked analysis outputs. It recomputes the prompt-order point estimates and the new active-first counts from shipped per-row tables, and an enclosed, separately tested archive recomputes the principal measurement, training and external-benchmark analyses from per-episode outcome tables. The prompt-order p-values and intervals and the 4B panel's statistics are regenerated from stored analysis outputs, not recomputed; the archive's README lists these cases. A clean-extraction test of the archive passed all checks. Protocols and amendments are summarised in Appendix F.

REFERENCES

Thierry Blankenstein, Jialin Yu, Zixuan Li, Vassilis Plachouras, Sunando Sengupta, Philip Torr, Yarin Gal, Alasdair Paren, and Adel Bibi. BiasBusters: Uncovering and mitigating tool selection bias in large language models, 2025. URL https://arxiv.org/abs/2510.00307.

Zhiyuan Fan, Tinghao Yu, Yuanjun Cai, Jiang Zhou, Jiangtao Guan, Jincheng Liu, Yun Yang, Dingxin Hu, Zhuo Han, Xing Wu, Feng Zhang, and Lilin Wang. Environment evolution for terminal agents, 2026.

Jiacheng Guo, Ling Yang, Peter Chen, Qixin Xiao, Yinjie Wang, Xinzhe Juan, Jiahao Qiu, Ke Shen, and Mengdi Wang. GenEnv: Difficulty-Aligned Co-Evolution between LLM agents and environment simulators, 2025.

Wanda Hou, Leon Zhou, Hong-Ye Hu, Yubei Chen, Yi-Zhuang You, and Xiao-Liang Qi. How focused are LLMs? A quantitative study via repetitive deterministic prediction tasks, 2025.

Miyoung Ko, Jinhyuk Lee, Hyunjae Kim, Gangwoo Kim, and Jaewoo Kang. Look at the first sentence: Position bias in question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1109–1121, 2020. doi: 10.18653/v1/2020.emnlp-main.84.

Stefan Krsteski and Charlotte Meyer. Predicting task difficulty without rollouts, 2026.

Bo Liu, Simon Yu, Yiding Jiang, Ao Qu, Andrew Zhao, Zichen Liu, Junsu Kim, Zijian Zhou, Seungone Kim, Tongzheng Ren, Mickel Liu, Hanfei Yu, Zhaorun Chen, Weiyan Shi, Paul Pu Liang, et al. SPADE: Self-Play in adaptive synthetic executable environments, 2026.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024. doi: 10.1162/tacl_a_00638.

Suyash Maniyar, Armaan Sandhu, and Abhishek Mishra. Measuring reward hacking and reasoning-answer decoupling under position-confounded optimization, 2026. URL https://arxiv.org/abs/2608.15445.

Fanzhe Meng, Guoxin Chen, Jiale Zhao, Shuang Sun, Zhiyu Lin, Wayne Xin Zhao, Ruihua Song, Ji-Rong Wen, and Kai Jia. CalibForge: Adversarial solver calibration for scaling learnable terminal tasks, 2026.

Pol G. Recasens, Alberto Gutierrez, Jordi Torres, Josep Ll. Berral, Javier Carnerero-Cano, Anisa Halimi, and Kieran Fraser. In-context bias propagation in LLM-based tabular data generation, 2025.

Jiaqi Shao, Hanck Chen, Wei Zhang, Maxm Pan, and Bing Luo. Do agent benchmarks measure capability? protocol validity in the age of agentic AI, 2026.

Wenhang Shi, Jinhao Dong, Yiren Chen, Zhe Zhao, Shuqing Bian, Wei Lu, and Xiaoyong Du. Scaling agentic capabilities via grounded interaction synthesis. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 4252–4263, 2026. doi: 10.1145/3770855.3817660.

Aivin V. Solatorio, Olivier Dupriez, and Rafael Macalaba. Field order should not matter: Permutation-invariant embedding model fine-tuning for structured metadata retrieval, 2026. URL https://arxiv.org/abs/2606.30473.

Michael Sullivan, Mareike Hartmann, and Alexander Koller. Procedural environment generation for tool-use agents. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 18544–18562, Suzhou, China, November 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.emnlp-main.936. URL https://aclanthology.org/2025.emnlp-main.936/.

Hiroki Tamba. Position bias is hidden behind ceiling effects: A permutation diagnostic for LLM benchmarks, 2026.

Xuemei Tang, Xufeng Duan, and Zhenguang G. Cai. Do large language models plan answer positions? Position bias in multiple-choice question generation, 2026.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel R. Bowman. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. In *Advances in Neural Information Processing Systems*, volume 36, pp. 74952–74965, 2023.

Xiao Wang. More thinking, more bias: Length-driven position bias in reasoning models, 2026. URL https://arxiv.org/abs/2605.06672.

Lorenz Wolf, Connor Watts, Roger Creus Castanyer, Geoffrey Bradway, Maxwill Lin, Augustine N. Mavor-Parker, and Matthew Daborn-Sargent. Breaking the solver bottleneck: Training task generators at the learnable frontier, 2026.

Xiaojun Wu, Cehao Yang, Honghao Liu, Xueyuan Lin, Zhichao Shi, Hao Zhou, Xuhui Jiang, Chengjin Xu, Jia Li, and Jian Guo. Envs-FORGE: Frontier-Optimized Reward-Grounded environment synthesis for agent RL, 2026.

Wang Yang, Chaoda Song, Xinpeng Li, Debargha Ganguly, Chuang Ma, Shouren Wang, Zhihao Dou, Yuli Zhou, Vipin Chaudhary, and Xiaotian Han. AgentCE-Bench: Agent configurable evaluation with scalable horizons and controllable difficulty under lightweight environments, 2026a.

Yiwei Yang, Haoxiang Zhang, Bingbing Wen, Yao Lu, Yuchen Wu, Lei Zhang, Julian McAuley, Pan Lu, and Bill Howe. Spurious tool use: When RL agents learn the wrong reason to act, 2026b. URL https://arxiv.org/abs/2609.16268.

Chujie Zheng, Hao Zhou, Fandong Meng, Jie Zhou, and Minlie Huang. Large language models are not robust multiple choice selectors. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=shr9PXz7T0.

Guojun Zhu, Xunheng Huang, Peng Yin, Jiahui Xie, Sanguo Zhang, and Doudou Zhou. Bad genius: Counterfactual-guided harness evolution beyond task-specific shortcuts, 2026. URL https://arxiv.org/abs/2609.18366.

A GENERATION PROMPTS AND THE PROMPT-ORDER CONDITIONS

Every generation request is an isolated call with one system message ("You design deterministic tool-use evaluation environments. You output ONLY a single JSON object and nothing else") and one user prompt. The prompt names the domain, requires exactly d sequential look-ups, gives a JSON skeleton, and states the rules: each non-terminal chain record lists its successor as the one "active" candidate among exactly two "superseded" decoys of the same kind; every candidate also appears as a full record, decoys being terminal; ids are unique and never leak a later record; sixteen kinds are listed; the answer is the sum of the chain's amounts; and each variant must differ from others in names, ids and amounts. The prompt-order conditions differ only in the `next_candidates` line of the skeleton:

O `[ {"kind": "...", "id": "...", "label": "...", "status": "active"}, ... EXACTLY 2 with "status": "superseded" ... ]`

P1 `[ {"kind": "ledger", "id": "3F9A1C77B2", "label": "ledger-4821", "status": "active"}, {... "8D2E5B04C1" ... "superseded"}, {... "A61F7E93D0" ... "superseded"} ]`

P2 the same three objects with the active one second; **P3**: with the active one third.

Condition O is byte-identical to the prompt that produced the fresh worlds. The prompts also tell the generator not to reuse the example ids. The design has 8 domains $\times$ requested depths $\{2, 6, 11, 16\}$ $\times$ 4 replicate blocks $= 128$ blocks per generator, with variant indices disjoint from the fresh pools; depth 1 is excluded because it has no branching decision.

Table 2: Prompt-order outcomes by condition (128 requests each). *Admitted*: structural and oracle admission. *Depth*: verified depth equals the requested depth. *Decoys*: every advertised gold-path decoy resolves. *First*: mean per-world share of branching decisions with the active candidate first, among admitted worlds. *Score*: mean frozen joint score over all 128 requests (inadmissible $= 0$).

	Sonnet 5				Terra				Luna			
	O	P1	P2	P3	O	P1	P2	P3	O	P1	P2	P3
Admitted	128	128	127	127	128	128	128	128	128	128	127	128
Depth	121	118	109	118	128	128	128	128	123	114	97	94
Decoys	128	128	127	127	128	128	128	128	118	119	123	128
First	0.715	0.793	0.343	0.370	1.000	1.000	0.033	0.007	1.000	1.000	0.771	0.191
Score	0.715	0.793	0.340	0.367	1.000	1.000	0.033	0.007	1.000	1.000	0.765	0.191

Analysis details. The randomisation test permutes the P1, P2 and P3 labels within each complete block (O held apart), with $p = (1+\#\{|T_{\text{sim}}| \geq |T_{\text{obs}}|\})/(100,000+1)$; the interval resamples blocks (10,000 draws). Holm was to be applied only if all three generators completed with valid integrity, which they did. The Sonnet 5 collection ran through Claude Code CLI 2.1.282 (the fresh pools used 2.1.278). One of its 512 first attempts ended locally with an empty output and an unknown provider outcome; one recovery request was made under a recorded policy, and its response (a valid P3 world that never lists the active candidate first) is used in the primary. Because it is not the original response, we bound its influence: excluding its whole block gives $T = 0.443$, and setting its lost score anywhere in $[0, 1]$ gives $T \in [0.436, 0.439]$, against $T = 0.439$ in the primary. The two Sonnet 5 failures are JSON syntax errors in P2 and P3; Luna's single failure is a chain whose active candidate does not resolve (P2). In total 2, 0 and 1 outputs were inadmissible for Sonnet 5, Terra and Luna.

B ACTIVE-FIRST FREQUENCY OF THE FRESH POOLS

This is a new descriptive analysis of stored data, made for this paper; no model was called. For every admitted fresh world we rebuilt the served world with the unchanged environment code and, along its gold chain, recorded the position of the active candidate in every record that advertises more than one candidate (a branching decision). A world enters if its frozen manifest marks it admitted and

the hash of its stored specification matches the manifest. The measure is the prompt-order study's per-world measure, applied to the native order.

Table 3: Active-first frequency of the fresh generated pools. Decision-level share pools all branching decisions; the world-level mean weights worlds with at least one branching decision equally, as in the prompt-order study. Worlds without a branching decision (mostly requested depth 1) carry no position information.

Generator	admitted worlds	with a branching decision	branching decisions	active first	share (%)	world-level mean
Sonnet 5	6,113	5,751	44,717	38,006	85.0	0.738
Terra	3,072	2,880	23,110	23,110	100.0	1.000
Luna	3,071	3,071	25,034	25,034	100.0	1.000

Sonnet 5's non-first placements put the active candidate second in 5,086 decisions and third in 1,625; 3,838 of its worlds list it first at every decision. Sonnet 5's share rises steeply with requested depth: 3.1% of decisions at depth 2, 22.4% at depth 3, 73.4% at depth 5 and 90.3% and 90.9% at depths 12 and 16. Deeper worlds have more decisions, which is why the decision-level share exceeds the world-level mean. An earlier audit of older pools from the same generators, written under the same specification rung, reported active-first fractions of 0.94–1.00 for the generators against 0.336 for the reference (Table 4).

Table 4: Older generated pools by specification rung (3,072 requests per pool; historical audit carried without recomputation). *Usable*: requests that materialise with a resolvable active chain. *Decoys resolve*: advertised superseded candidates that resolve. *Depth@16*: mean realised depth at requested depth 16. *Branching*: non-terminal records with more than one candidate. *Active first*: among branching records, the share listing the active candidate first. PERT is an authored control, one mechanically extended seed per domain.

family	rung	usable	decoys resolve	depth@16	branching	active first
REF (reference)	—	1.000	1.0	16.0	1.000	0.336
PARA (paraphrased reference)	—	1.000	1.0	16.0	1.000	0.336
PERT (authored, unshuffled / shuffled)	—	1.000	1.0	16.0	1.000	1.000 / 0.339
gpt-5.6-luna	1 permissive	0.999	0.115	14.7	0.959	0.9998
gpt-5.6-luna	3 pinned+records	1.000	0.993	16.0	1.000	1.000
gpt-5.6-terra	1 permissive	1.000	0.003	16.0	0.925	0.9999
gpt-5.6-terra	3 pinned+records	1.000	0.9998	16.0	1.000	1.000
claude-sonnet-5	1 permissive	0.998	0.0003	16.0	0.607	0.991
claude-sonnet-5	3 pinned+records	0.997	0.9996	15.9	1.000	0.936
any shuffled generated arm	3	—	—	—	1.000	0.332–0.338

C MEASUREMENT DETAILS

Populations. The Sonnet 5 pool has 6,144 requests, 6,113 admitted; Terra 3,072, all admitted; Luna 3,072, 3,071 admitted. Each presentation was solved once. The shuffled presentation uses the stored permutation with seed 20261520. The 1.7B paired intervals use 5,000 bootstrap draws stratified by domain and requested depth; equal weighting of the 128 domain-by-depth cells changes no generator's gain by more than 0.02 points. The 4B panel selected eight worlds per domain and requested depth from each pool by a salted hash rank, fixed before any 4B outcome and verified against the frozen manifests; the solver was served with vLLM at the checkpoint's own precision and chat template, with the same turn budget and a generation budget of $256+128\,\max(d_{\text{requested}}, d_{\text{verified}})$ tokens.

The conditional depth test. Among discordant worlds of a generator, let N_w indicate that the native presentation won. Within each domain k, the test conditions on the number of native wins and asks whether native wins are associated with a depth score $s(d)$ that combines a linear and a quadratic term in verified depth, both centred within the domain. The statistic Q is a quadratic form in the domain-summed scores $\sum_k \sum_{w \in k} (N_w - \bar{N}_k)\, s(d_w)$, and its null distribution comes from 50,000 conditional re-allocations of native wins within domains. A constant domain-specific preference for native order leaves Q unchanged. The test is two-sided and detects departures in either direction; it does not assert a monotone law. The within-domain secondary statistic Q_{dom} sums per-domain quadratic forms, so it can detect domain patterns that cancel when pooled; it has

its own Holm family and never replaces the primary. For Sonnet 5, the native-win shares among discordant worlds in depth bands 1–5, 6–10 and 11–16 are 0.655, 0.869 and 0.803, a rise and fall that is descriptive, not a monotone law.

Table 5: The 1.7B measurement on fresh worlds (E1) and a corrected repeat on older worlds (F). *Native/shuffled*: traversal completion. Paired gain: native minus shuffled, points, 95% world-cluster intervals. *Disc.*: discordant worlds, with native (N) and shuffled (S) wins. Q and raw p: the conditional depth test (a raw p printed as $\leq$ is at the Monte Carlo resolution). Holm: across the three generators within each study; the control lies outside the family. F's Sonnet 5 worlds came from the direct API, E1's through the command-line interface.

		worlds	native	shuffled	gain, pp [95%]	disc. (N/S)	Q, raw p	Holm p
F	Sonnet 5	3,063	0.932	0.849	$+8.29$ [7.12, 9.44]	398 (326/72)	16.14, 2.6×10^{-4}	7.8×10^{-4}
	Terra	3,072	0.921	0.782	$+13.90$ [12.66, 15.14]	473 (450/23)	0.65, 0.731	1.000
	Luna	3,072	0.968	0.864	$+10.38$ [9.28, 11.52]	365 (342/23)	0.24, 0.893	1.000
	REF/REFS	3,072	0.871	0.870	$+0.16$ [-1.40, 1.66]	621 (313/308)	5.13, 0.078	—
E1	Sonnet 5	6,113	0.943	0.879	$+6.36$ [5.59, 7.17]	681 (535/146)	18.92, 6.0×10^{-5}	1.8×10^{-4}
	Terra	3,072	0.925	0.773	$+15.23$ [13.96, 16.50]	516 (492/24)	0.11, 0.950	0.950
	Luna	3,071	0.963	0.845	$+11.76$ [10.58, 12.96]	413 (387/26)	4.86, 0.084	0.169
	REF/REFS	3,072	0.861	0.854	$+0.65$ [-0.88, 2.18]	618 (319/299)	1.60, 0.451	—

Table 6: Pre-declared sensitivities of the E1 depth test (Holm within each row group). No decision changes. Luna's requested-depth result, Holm $p = 0.0514$, has a Monte Carlo standard error of about 0.0014, so it sits at the resolution of the allocations and supports no conclusion either way.

sensitivity	generator	excluded	Q	raw p	Holm p
primary (full population)	Sonnet 5	0	18.92	6.0×10^{-5}	1.8×10^{-4}
	Terra	0	0.11	0.950	0.950
	Luna	0	4.86	0.084	0.169
unknown-outcome recoveries excluded	Sonnet 5	4	18.90	4.0×10^{-5}	1.2×10^{-4}
	Terra	0	0.11	0.948	0.948
	Luna	0	4.86	0.087	0.173
verified deeper than requested excluded	Sonnet 5	62	20.44	4.0×10^{-5}	1.2×10^{-4}
	Terra	70	0.11	0.950	0.950
	Luna	672	5.65	0.056	0.112
on-depth worlds only	Sonnet 5	908	22.27	$\leq 2.0 \times 10^{-5}$	$\leq 6.0 \times 10^{-5}$
	Terra	70	0.11	0.950	0.950
	Luna	680	5.65	0.058	0.117
requested-depth axis	Sonnet 5	0	18.74	8.0×10^{-5}	2.4×10^{-4}
	Terra	0	0.03	0.986	0.986
	Luna	0	7.26	0.026	0.051

Second-solver sensitivities. Excluding worlds with unresolved advertised decoys, identified by generator pool and item, removes 0, 2 and 51 worlds and gives gains of $+3.43$ [1.57, 5.30], $+12.33$ [10.47, 14.29] and $+2.57$ [0.41, 4.73] points, with raw exact $p = 8.2 \times 10^{-4}$, 8.5×10^{-28} and 0.0504 (no multiplicity adjustment). An earlier implementation of this sensitivity matched exclusions by bare item names, which collide across pools; it was corrected before these results were finalised, and only the corrected values are reported. The 4B strict-success control is -1.07 [-2.05, -0.20] points, raw $p = 0.052$. For the 1.7B solver on the same subsample, the native gains are $+5.8$, $+16.9$ and $+12.6$ points (sign-test p for the difference of differences: 0.157, 0.013, 4.5×10^{-8}). The 4B gain minus the 1.7B gain is -4.5 points [-7.4, -1.6] for Terra and -9.1 [-12.0, -6.2] for Luna; for Sonnet 5, -2.4 [-4.9, 0.1], it is indeterminate.

Strict success of the 1.7B solver. Correct final answers are rare for this solver in both presentations: 225 and 209 of 6,113 Sonnet 5 worlds (native, shuffled), 179 and 174 of 3,072 Terra worlds, 55 and 45 of 3,071 Luna worlds, and 175 and 168 of 3,072 reference worlds (REF, REFS). No test was declared for them.

Response scales. A descriptive shape parameter α from the accumulation model $P(\text{success} \mid d) = \exp[-\beta_0 \alpha^{d-1} d]$ (Hou et al., 2025) was pre-declared as a supporting summary. It orders the generators differently from the conditional test (Table 7): its native-minus-shuffled contrast is -0.015 $[-0.043, 0.012]$ for Sonnet 5, whose test rejects, and $+0.074$ and $+0.054$, with intervals excluding zero, for Terra and Luna, whose tests do not. The two describe different response scales, so any claim about the "shape" of an order effect has to name its scale. The declared paraphrase floor $|\alpha_{\text{PARA}} - \alpha_{\text{REF}}|$ is 0.037 [0.013, 0.061], and every generated arm's margin over it is indeterminate.

Table 7: Descriptive shape contrasts (bootstrap intervals) beside the paired gains and the depth-test Holm p.

study	pool	$\alpha_{\text{nat}} - \alpha_{\text{shuf}}$ [95%]	gain (pp)	Holm p
F	Sonnet 5	$+0.147$ [0.105, 0.196]	$+8.29$	7.8×10^{-4}
	Terra	$+0.057$ [0.031, 0.089]	$+13.90$	1.000
	Luna	-0.006 [-0.052, 0.044]	$+10.38$	1.000
	REF−REFS	$+0.004$ [-0.027, 0.034]	$+0.16$	—
E1	Sonnet 5	-0.015 [-0.043, 0.012]	$+6.36$	1.8×10^{-4}
	Terra	$+0.074$ [0.045, 0.105]	$+15.23$	0.950
	Luna	$+0.054$ [0.010, 0.103]	$+11.76$	0.169
	REF−REFS	$+0.016$ [-0.013, 0.044]	$+0.65$	—

The wording comparison. PARA serves REF worlds with instructions rewritten by `gpt-5.6-luna`. On 3,069 matched worlds it completes 0.713 against 0.861 for REF; 654 worlds were won only by REF and 201 only by PARA, and PARA fails mainly by stopping early. This comparison was exploratory and made after outcomes, and PARA ran in a separate solver session, so a session effect is not excluded. An outcome-blind review of 128 sampled rewrites using two LLM reviews with LLM adjudication found no failed requirement but 28 rewrites ambiguous about whether the sum covers the chain or every record met; a deterministic screen of the whole pool then flagged 679 such rewrites and 3 invalid ones. Excluding those 682 worlds gives a floor of 0.031 [0.006, 0.057]. Traversal grading ignores the numeric answer, but an instruction change can still change the trajectory.

D TRAINING DETAILS

The six fits used full-parameter AdamW (learning rate 1×10^{-5}, batch 16, 18 warm-up steps, 3 epochs, 576 updates, FP32 master weights with BF16 compute), a maximum sequence length of 16,384 tokens and 2,505,120 supervised tokens per fit. All 310 weight tensors changed in every fit. Before any fit, a pilot found that tool responses were serialised differently in training and serving; the prefix-preserving template adopted for both was verified to match on 35,169 decisions. The teacher-forced probe presents 256 reference worlds with the active candidate placed first, second or third, and reads the first candidate each checkpoint names (5,376 calls). The episode categories in Figure 4b come from a replay of the stored traces made after the studies (no model call); a separate verification script replayed all seven checkpoints' REF traces (21,504 episodes) and obtained identical category counts. The replay identifies which candidate was named and what was answered, not why.

Table 8: Traversal and strict success (%) of each checkpoint, equal cell weights. Sonnet-N/S: fresh Sonnet 5 worlds in native and shuffled order (secondary).

checkpoint	REF	REFA	Sonnet-N	Sonnet-S	REF strict	Sonnet-N strict
17 NATIVE	13.0	100.0	77.8	21.5	10.3	19.9
17 RANDOM	99.8	100.0	97.4	97.0	21.3	29.1
29 NATIVE	41.3	100.0	93.2	61.6	14.7	26.9
29 RANDOM	100.0	100.0	99.3	99.4	23.2	30.4
43 NATIVE	14.0	100.0	75.4	23.2	10.5	19.0
43 RANDOM	100.0	100.0	97.8	98.0	22.4	30.6
BASE	91.7	99.8	96.9	94.5	0.0	0.1

Table 9: Teacher-forced first decision: of 256 worlds, how often each checkpoint names the active candidate at each position, and which decoy it names when the active candidate is third.

checkpoint	active at 1	active at 2	active at 3	at 3: 1st-listed decoy	at 3: 2nd-listed decoy
17 NATIVE	256	105	0	116	140
17 RANDOM	256	255	256	0	0
29 NATIVE	256	241	16	7	233
29 RANDOM	256	256	256	0	0
43 NATIVE	256	90	0	128	128
43 RANDOM	256	256	256	0	0
BASE	256	210	28	78	150

E EXTERNAL BENCHMARK

AgentCE-Bench (Yang et al., 2026a) was run at a pinned revision on 512 fresh worlds over six domains and horizons 1, 5, 11 and 21 with two fixed solvers (`Qwen3.5-9B`, `Granite-4.2-8B`), for 2,048 scheduled episodes, all analysed. The stored presentation is the official query unchanged; the shuffled presentation runs the same query, filtering and truncation and then permutes the returned records, freshly per query, so records, flags and budget charges are identical. The primary is a conditional horizon-shape test with Holm across the two solvers; a sign-flip level test forms a separate secondary family (Holm $p = 0.802$). Granite's rejection lies about 1.9 Monte Carlo standard errors below 0.05 (an exact enumeration gives Holm $p = 0.0478$), and 7 of its 24 domain-by-horizon cells complete nothing in either order (Table 10 pools the cells by horizon).

Table 10: External benchmark: completions by horizon and shuffled minus stored completion with 95% stratified world-bootstrap intervals. The completion counts are pooled over domains, whereas the contrasts weight domains equally within each horizon and all domain-by-horizon cells equally overall; subtracting the pooled proportions therefore does not exactly reproduce the contrasts.

solver	horizon	stored completed	shuffled completed	shuffled − stored, pp [95%]
Qwen3.5-9B	1	109/126	111/126	$+1.59$ [-4.76, 7.94]
	5	83/128	81/128	-1.59 [-10.25, 7.00]
	11	40/129	39/129	-0.72 [-10.97, 9.49]
	21	11/129	16/129	$+3.93$ [-3.03, 10.90]
	all	243/512	247/512	$+0.80$ [-3.19, 4.79]
Granite-4.2-8B	1	96/126	86/126	-7.94 [-13.49, -2.38]
	5	65/128	68/128	$+2.34$ [-3.03, 7.65]
	11	7/129	10/129	$+2.27$ [-0.83, 5.45]
	21	0/129	0/129	$+0.00$ [0.00, 0.00]
	all	168/512	164/512	-0.83 [-2.80, 1.32]

F HISTORY, AMENDMENTS AND RESOURCES

Order of events. (1) An exploratory study served older generated, authored and reference worlds to the same 1.7B solver in five sessions; it produced the original observation that the generators list the active candidate first (Table 4). (2) A review found defects in its uncertainty and trace analyses: a bootstrap that broke the pairing of a world's presentations, a hazard analysis with a wrong risk set, and pooled p-values that treated shared worlds as independent. These were corrected retrospectively on the same data and the affected claims were withdrawn. The retrospectively corrected estimates are not used as evidence in the main text; the corrected repeat (F) cited there is a separate, prospective re-run on those older worlds. (3) A written plan then fixed questions, estimands, multiplicity families, grids, splits and the training recipe before any new outcome, and each later study froze its configuration with input and code hashes before its outcomes: the corrected repeat on older worlds (F, 30,720 rows), the fresh-world measurement (E1), the matched training (ET), the external benchmark (E2) and, later, the prompt-order experiment and the 4B panel. The prompt-order and 4B studies were follow-ups designed after the E1 results; the 4B solver had been identified by a competence screen before the fresh-world study and was never selected for an order effect.

Amendments and deviations. ET adopted the prefix-preserving template and FP32 master weights after pilot evidence and before any full fit. E2 replaced a comparator that shuffled before filtering, which changed which records were visible, before any main outcome. In E1, a defect in sensitivity exclusions keyed by bare item names was fixed before any Terra or Luna outcome, and a staging function added to the deployment code after the freeze was not recorded at the time; it changed no existing function and is disclosed here. The fresh Sonnet 5 collection had one refusal with verified zero usage and four first attempts with unknown provider outcomes during a local power loss; these were recovered, and a pre-declared sensitivity excludes the four unknown-outcome recoveries (Table 6). The prompt-order collection had one comparable recovery (Appendix A).

Scope. A held-out calibration study, mechanism controls and a paid-model solver panel were optional in the plan and were not run; reinforcement learning, recovery training and mixed-order training were outside its scope. A follow-up training study with partially ordered data is not reported here.

Resources. Recorded consumption through the prompt-order and 4B studies was $241.27 (GPU $186.76, API $47.43, CPU estimates $7.09), as list-price estimates rather than settled invoices; the Sonnet 5 generations ran under an included subscription.

G SUPPLEMENTARY ARCHIVE

The anonymous archive has two parts. `main2/` rebuilds this paper's numbers and figures: `build/` holds the unchanged checkpoint-18 evidence library and number generator, the main2 number and figure generators and both source registries; `data/` holds every registered analysis output with its SHA-256 (the prompt-order and 4B analyses, the ET trace audit and its independent verification, the new active-first analysis and the Figure 1 world). Its `reproduce.py` checks each source hash, regenerates every macro, table, figure-data file and evidence-index record, and compares them with the shipped copies; it also recomputes the active-first counts of Appendix B from a shipped per-world position table and the prompt-order primary contrasts, bounds and admission counts from the shipped per-request score table. `cp18-artifact/` is the earlier tested archive, unchanged, which recomputes the F, E1, ET and E2 principal analyses from compact per-episode outcome tables. The 4B panel's statistics are regenerated from its stored analysis output and were recomputed independently with a separate verification script from the solver journals; the journals, raw generator responses, generated worlds, trajectories and the six fine-tuned checkpoints are not shipped. Each evidence-index record gives the source file, SHA-256, JSON pointer and formatting rule of one number in the paper.