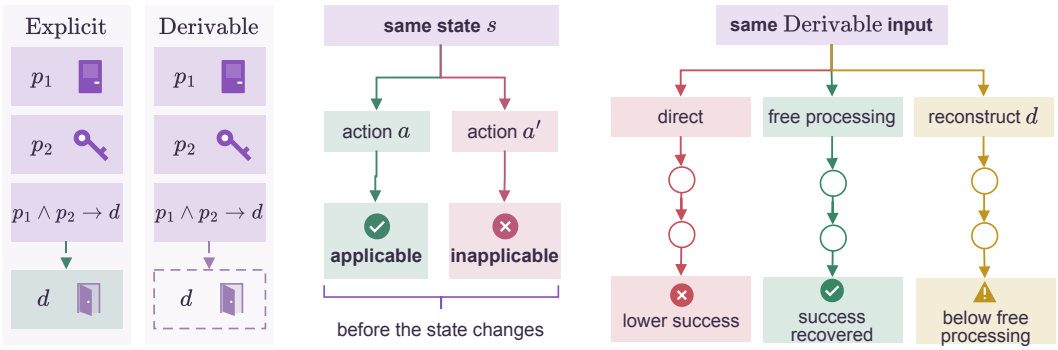
LOGICALLY REDUNDANT, BEHAVIORALLY DIFFERENT: OMITTING DERIVABLE INFORMATION IN LLM AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

LLM agents use tools over multiple steps, accumulating observations, tool outputs, and memory as they act. Existing work therefore compresses or omits observations and interaction history to manage growing context. Logically, information derivable from what remains can be omitted without loss. However, how such omission affects agent behavior remains underexplored. This paper uses paired *Explicit* and *Derivable* conditions to study this question in multi-step tool-use tasks. *Explicit* provides selected derived information directly in the current state representation, while *Derivable* omits the same information only when it remains exactly derivable from retained facts and rules. In TextWorld, *Explicit* improves task success by 61.7 percentage points, completing 58/60 tasks compared with 21/60 under *Derivable*. The effect grows as more derived conclusions are stated, is not reproduced by repeating retained premises, and also appears in next-action choices from the same environment state. The difference weakens or disappears when additional reasoning or preparation is available before action and also varies across executors. Derivability alone does not guarantee behavioral preservation. Whether omission preserves behavior depends on the executor, inference settings, and pre-action processing.



(A) Derivable omission. (B) Same-state action. (C) Pre-action processing.

Figure 1: Why derivable information can still change behavior.

1 INTRODUCTION

LLM agents act through tools over multiple steps (Yao et al., 2023), accumulating observations, tool outputs, and memory as they act. As this context grows, systems compress interaction history (Kang et al., 2026) or omit task-irrelevant observations (Kerboua et al., 2026). These methods determine not only what remains visible to the model, but also what must be derived from retained information before the next action.

Information derivable from retained facts and rules is a natural candidate for omission when structured state stores recompute derived views on demand or tools return raw state without system-computed derived fields. Logically, such information can be omitted without loss when it remains exactly derivable. Omission does not remove the derivation, however. It moves the derivation from 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.

system to the model, which must derive the conclusion from those facts and rules when choosing what to do next. Theories of usable information (Xu et al., 2020) and chain-of-thought expressivity (Li et al., 2024) suggest that this shift can matter for models with limited computation. How it affects the behavior of multi-step agents remains underexplored.

This paper studies this gap by comparing two ways of presenting the current state in multi-step tool-use tasks. *Explicit* provides selected task-relevant derived information directly, while *Derivable* omits the same information only when it is exactly derivable from retained facts and rules. Figure 1A illustrates this distinction with a simple door-and-key example. Both conditions retain the facts and rule supporting the derived conclusion d, which is directly shown only in *Explicit*. The task, tools, environment, and common model-visible content are otherwise unchanged. Each request is built from the current state, so the comparison isolates the current-state interface from the compression of interaction history. Additional controls examine the content provided to the model and the processing allowed before action, distinguishing derived conclusions from extra text and repeated premises, and comparing different ways of preparing from the same *Derivable* input.

When the executor has limited computation before action, omitting derivable information can substantially reduce task success. In TextWorld, Qwen3.8-27B with thinking disabled completes 58/60 tasks under *Explicit* and 21/60 under *Derivable*, and the same configuration completes 16/100 and 1/100 ALFWorld tasks under the two conditions. Success also increases as more derived conclusions are stated, while neutral text and repeated premises do not reproduce the benefit. The difference weakens or disappears when more computation is available before action and also varies across executors.

Further analyses move from task success to the next action chosen from a shared environment state (Figure 1B). In TextWorld, every first divergence that changes action applicability favors *Explicit*. On pre-selected ALFWorld states, *Explicit* also yields more applicable next actions for Qwen and Gemma, so the difference does not require earlier divergence in the trajectory. Starting from the same *Derivable* input (Figure 1C), free-form preparation before action restores task success, while requiring the model to reproduce the complete derived state is less effective under the same preparation budget.

This paper makes three contributions.

- It defines paired *Explicit* and *Derivable* conditions in which omitted derived information remains exactly derivable from retained facts and rules throughout task execution.
- It shows that omission can change both task completion and the next action chosen from the same environment state even when the omitted information remains exactly derivable, with content controls distinguishing derived conclusions from neutral text and repeated premises.
- It shows that the effect depends on the executor and on the computation available before action. Additional pre-action processing can reduce or eliminate the difference in some settings, while requiring complete reconstruction remains less effective under the same preparation budget.

The interface choice also creates a systems trade-off. In the TextWorld preparation sample, omitting derived information reduces the mean first prompt from 5,022 to 4,720 tokens. Free-form preparation restores task success to 60/60, while system-side deterministic derivation completes 58/60 tasks with about 60% of the tokens and half the calls.

2 Background and Related Work

Context reduction is already a mature design space in LLM systems. LLMLingua compresses prompts to reduce inference cost while preserving useful downstream performance (Jiang et al., 2023). Agent systems apply the same objective to interaction context. ACON compresses observations and interaction history in long-horizon tasks (Kang et al., 2026), while FocusAgent removes task-irrelevant parts of Web observations while largely retaining performance (Kerboua et al., 2026). Decision-aware memory methods further select context according to its estimated effect on future actions and outcomes (Guan et al., 2026). Current methods therefore focus on learning what to retain and how to compress it, while leaving open when information that remains derivable can be omitted without changing behavior.

However, successful context reduction does not imply that information left available will always be used robustly. Lost in the Middle shows that long-context performance varies with the position of relevant information (Liu et al., 2024). LLMs Get Lost shows that distributing task information across multiple turns can substantially change model performance (Laban et al., 2026). Semantics-preserving transformations can also destabilize agentic responses (de Zarzà et al., 2026). These findings show that availability and semantic preservation do not guarantee stable use. The case of omitting information that remains derivable from retained facts and rules has received much less attention.

One important form of this sensitivity appears when a model must derive a conclusion that is not provided directly. The Reversal Curse shows that relational facts learned in one direction may not support the reverse query (Berglund et al., 2024). MixRea benchmarks explicit and implicit reasoning formulations and reports substantial inconsistencies between them (Cai et al., 2026). Before the Action removes explicit conclusions while retaining the supporting facts to evaluate prospective hypothesis discovery (Zhong et al., 2026). Rule-reasoning benchmarks evaluate derivation from stated facts and rules at increasing depths (Clark et al., 2020; Tafjord et al., 2021), and models often fail to compose reasoning steps that they can perform separately (Press et al., 2023; Dziri et al., 2023).

Theory explains why such effects can arise. For a predictor with limited computation, a deterministic transformation of the input can increase the information it can use (Xu et al., 2020), and intermediate reasoning steps can extend the computations available to a fixed-depth transformer (Feng et al., 2023), including inherently serial problems (Li et al., 2024). A stated conclusion can therefore help a model with limited computation. Classical planning exposes a related distinction, where eliminating derived predicates can require growth in plan length or representation size (Thiébaux et al., 2005). Related approaches organize intermediate computation before producing a response, for example by separating planning from execution (Wang et al., 2023) or by delegating intermediate programs to an interpreter (Gao et al., 2023). This paper tests the resulting behavioral effect in multi-step agents while verifying that the omitted information remains exactly derivable.

At the system level, this raises the question of what is formally guaranteed when information is removed. Contract-based design (Benveniste et al., 2018), verified structured updates over projected views (Bai & Cai, 2026), runtime rule enforcement for agents (Wang et al., 2026a), compiled workflow verification (Wang et al., 2026b), and temporal verification of stateful tool-enabled deployments (Mercado & Lomuscio, 2026) can certify that a representation, update, or execution satisfies a specified property. Such guarantees do not determine whether a learned agent will preserve its action choices after information is omitted under a particular execution protocol.

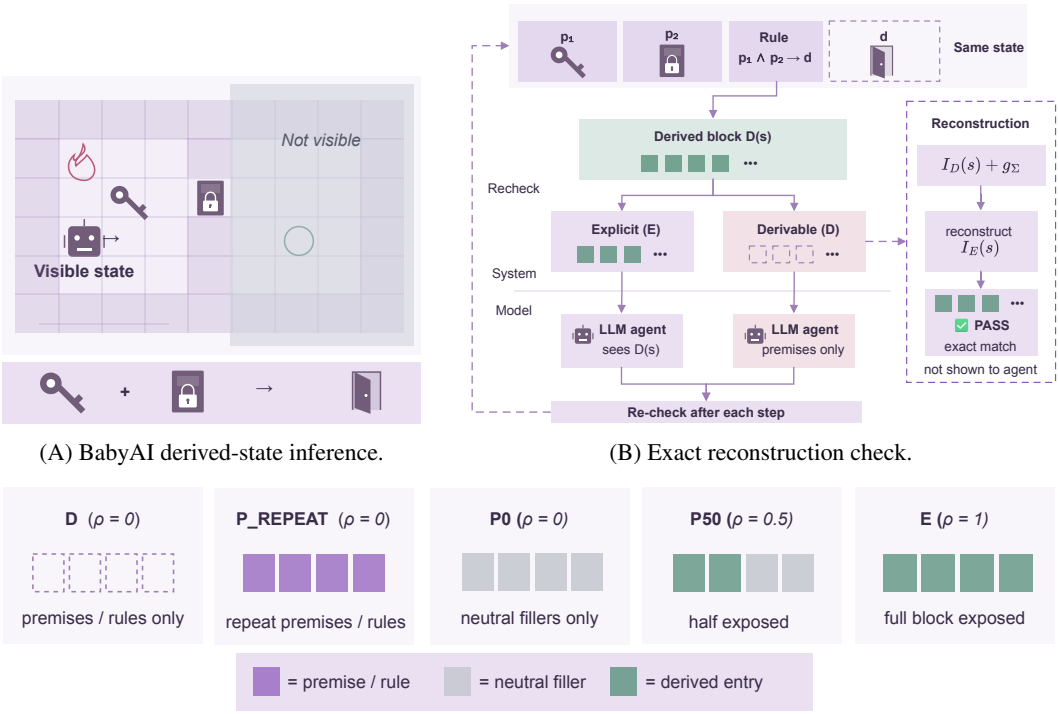
The behavioral consequences of omitting derivable information therefore remain underexplored in multi-step tool use. This paper examines omissions of derived state that remains exactly derivable from retained facts and rules, and tests how their effect depends on the executor and on the processing that precedes action.

3 Reconstructible Derived-State Interfaces

3.1 Core Interface Construction

Stateful agent interfaces can compute task-relevant intermediate conclusions from the current environment and a public rule set before the model is invoked. The intervention varies whether this derived-state inference is performed by the system and written into the model interface. Here, derived-state inference is deterministic rule evaluation and requires no auxiliary model call. Explicit includes the resulting derived state. Derivable keeps the same task, tools, current environment state, and public rules, but omits those conclusions. The unmodified Derivable condition is denoted D below. At each step, the agent uses the current observations and public memory to select an action. The resulting state determines the facts and derived conclusions available at the next decision.

For illustration, Figure 2A uses a task from BabyAI. The mission is *open the blue door, then go to the green ball*. At the illustrated initial state, the agent-visible memory contains a locked blue door at `(4,3)` and a blue key at `(2,4)`. Under the public rule registry, these visible objects satisfy the matching-door and matching-key conditions and yield the derived conclusion *barrier identity resolved*. At this state, Explicit differs from Derivable by the complete 12-entry `derived` block,



(A) BabyAI derived-state inference.

(B) Exact reconstruction check.

(C) Content controls.

Figure 2: Reconstructible interfaces and content controls.

including a positive *barrier identity resolved* entry. The mission, visible state, public rules, and tools are shared, while Derivable omits this block in full. Here, the conclusion identifies the barrier from the observed door and key, after which the agent still has to choose the actions that open it. As the task progresses, new observations and actions change the current facts and their derived conclusions, so the relevant derived state must change with the environment.

3.2 Exact Reconstruction in Stateful Tasks

Let $F(s)$ denote the model-visible fields shared by the two interfaces at state s, and let $D(s)$ denote the preselected task-relevant derived-state block. Let the fixed renderer $\mathcal{R}$ serialize these fields into the actual model input. The two interfaces are

$$I_E(s) = \mathcal{R}(F(s), D(s)), \qquad I_D(s) = \mathcal{R}(F(s)). \tag{1}$$

Figure 2B shows the exact reconstruction check applied before behavioral evaluation. Let Σ denote the fixed public rule set, and let g_Σ be the deterministic reconstruction function that uses those rules to rebuild the complete Explicit interface from the Derivable interface. A task–interface pair is included in the behavioral comparison only when

$$g_\Sigma(I_D(s)) = I_E(s). \tag{2}$$

The reconstructed interface is used only for this check and is never supplied to the agent. The check reads only what remains in Derivable together with the fixed public rule set. The shared action catalog specifies the available operations and their public preconditions. This check establishes derivability relative to the public rule set Σ, its declared semantics, and the deterministic evaluator g_Σ. The TextWorld D and E action requests do not state the least-fixed-point or closed-world procedure explicitly, while the C and R preparation calls receive it. Whether the two interfaces preserve behavior is evaluated separately.

State changes make exact reconstruction a continuing requirement throughout execution. Agent actions change the environment, and the relevant derived state can change with it. After each legal state transition, the system-side interface builder recomputes the new $D(s')$ with the same deterministic rule evaluator, and the reconstruction checker repeats the exact-match test for the new state. This process is applied state by state over the evaluated execution horizon, so reconstructibility is reverified throughout closed-loop execution. The validation suite covers observation-dependent branching, irreversible actions, and changing derived conclusions. Appendix A presents the complete reconstruction procedure, worked examples, and validation results.

3.3 Interface and Preparation Controls

Figure 2C constructs two families of interface controls from the Derivable baseline D. P_REPEAT repeats task-relevant premises or public rules already retained in D without adding a new derived conclusion, testing whether another presentation of the retained premises can reproduce the Explicit–Derivable difference.

A second family holds an additional text block fixed while varying the derived content stated within it. P0 contains only neutral padding text. The D–P0 comparison measures the effect of adding this block. P50 exposes a prespecified half of the target derived entries, and E exposes the complete target derived block. Let $X(s) \subseteq D(s)$ denote the target entries stated in this additional block and define

$$\rho(s) = \frac{|X(s)|}{|D(s)|}. \tag{3}$$

P0, P50, and E correspond to $\rho = 0$, 0.5, and 1, respectively. P50 uses one prespecified subset, so ρ indexes this fixed exposure series. The P0–P50–E progression examines how increasing the amount of stated derived information affects task performance.

A further question is whether changed pre-action processing can alter behavior from the Derivable interface. Guided Preparation (C) and Mandatory Reconstruction (R) receive the Derivable input together with an explicit public reconstruction procedure and the same preparation budget. The D and E action requests do not include that procedural text. C allows free-form working notes, while R requires the complete derived state to be reconstructed. C passes its notes to the subsequent action request, while R places its generated state in the same derived-state field used by E.

The C–R comparison tests how the preparation objective matters once the same procedure and budget are provided. Comparisons of C or R with D alone do not isolate an additional model call from the added procedural text, so a separate control adds an adapted description of the procedure to D without a preparation call. Explicit supplies the correct derived state as a reference for model-generated reconstruction.

4 Experimental Setup

Benchmarks. Experiments use TextWorld (Côté et al., 2019), BabyAI (Chevalier-Boisvert et al., 2019), and ALFWorld (Shridhar et al., 2021). The TextWorld tasks use a symbolic interface with Boolean facts, public rules, and opaque operation identifiers with explicit preconditions and effects. Because actions change which operations are available later, the task-relevant derived state evolves throughout an episode. BabyAI introduces spatial action under partial observation, where the agent must act from a compact symbolic view of the local scene and task. ALFWorld uses household manipulation tasks grounded in PDDL, combining longer action sequences with explicit preconditions and goal conditions. The three environments vary the form of state, the length of interaction, and the kinds of decisions for which derived information may be useful.

Models. Experiments use Qwen3.8-27B in BF16 and FP8, DeepSeek-V4-Flash, GLM-5.3-Flash, and Gemma 4 31B IT. Qwen is the main executor for the TextWorld content and preparation comparisons, which use thinking disabled. To vary the computation available before action, Qwen is also evaluated with thinking enabled in TextWorld and at two generation limits in BabyAI. DeepSeek and GLM are evaluated on the same BabyAI tasks, and Qwen, Gemma, and DeepSeek are compared on the same ALFWorld states, with task completion evaluated for Qwen and DeepSeek on the corresponding tasks. Appendix B.2 lists the complete settings.

Comparisons. Section 3.3 defines the conditions used below. Explicit and Derivable test whether omission changes behavior when the missing conclusions remain exactly derivable. Content controls examine whether the benefit comes from the derived conclusions or from additional text and repeated premises. Preparation controls examine whether the same retained information can support better decisions when the model processes it before acting. Same-state comparisons locate differences in the next action before the paired runs cause different changes to the environment.

Evaluation. TaskPass is the proportion of tasks completed within the allowed number of actions. Each paired task comparison starts from the same task and initial state. Same-state analysis compares the actions chosen from an identical environment state and records whether their preconditions are satisfied. In TextWorld, an episode ends at the first inapplicable action, and Appendix C also reports a variant that continues after rejected actions. Appendix B provides the task samples, model settings, and statistical procedures.

5 RESULTS

This section compares task completion across environments and executors, examines the effects of derived content and preparation, and analyzes action choices from the same environment state.

5.1 BEHAVIORAL EFFECTS

Table 1 summarizes task completion across benchmarks and executors. With thinking disabled, Explicit raises Qwen BF16's TaskPass in TextWorld from 35.0% to 96.7%, an increase of 61.7 percentage points (95% CI [50.0,73.3]). Although the retained facts and rules determine the omitted conclusions exactly, leaving those conclusions implicit substantially reduces task completion. In a separate comparison of Explicit with premise repetition, continuing after rejected actions gives the same 65.0-point difference as stopping at the first inapplicable action, so the termination rule does not account for the exposure effect in that comparison (Appendix C). The same Qwen configuration also shows a task-level effect in ALFWorld, completing 16/100 tasks under Explicit and 1/100 under Derivable, a difference of 15.0 percentage points (95% CI [8.0,23.0]). Absolute ALFWorld success remains low under both conditions (Appendix C.10).

The difference is smaller or absent in settings that allow more computation before action. On the 60 TextWorld tasks of Figure 3B, Explicit exceeds neutral padding by 46.7 percentage points with thinking disabled, while all three conditions complete every task with thinking enabled. In BabyAI, where Qwen uses thinking enabled, Explicit completes 15 percentage points more tasks at the 8K generation limit, and both conditions complete 55/60 tasks at the 16K limit (Section 5.2). The difference also depends on the executor. On the same BabyAI tasks, DeepSeek and GLM complete nearly all tasks under Derivable and have reverse-signed point estimates, as does DeepSeek in ALFWorld task completion, although exact paired tests leave these reverse-signed estimates uncertain.

5.2 CONTENT, PREPARATION, AND BUDGET CONTROLS

Figure 3A examines how task success changes with the amount of derived information provided. Performance improves at both steps of the prespecified P0–P50–E series, while mean first-prompt length decreases from 5,135 to 5,103 to 5,071 tokens (Appendix B.5). Adding neutral padding alone produces only a small change from D. The increase therefore tracks the selected derived content and is not explained by prompt length alone. Because P50 uses one prespecified subset, this series does not separate the amount of exposed content from the identity of the exposed conclusions.

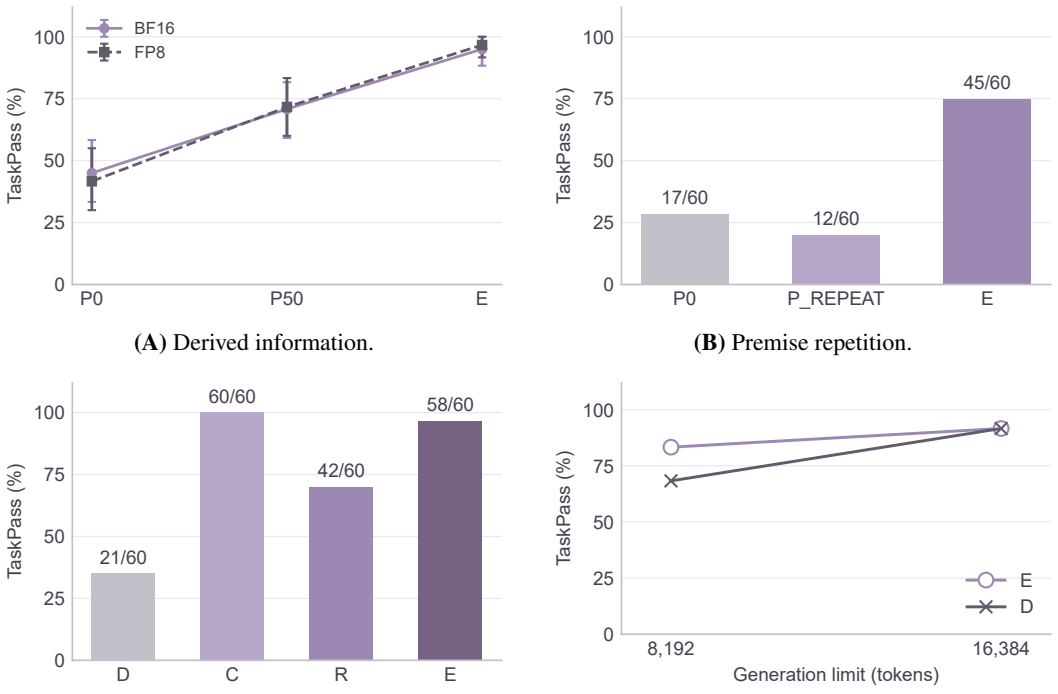
The repetition comparison separates this content effect from another presentation of the supporting facts and rules (Figure 3B). P_REPEAT performs below P0 in the observed results, and both remain well below E. Repetition leaves the model to derive the conclusions even after their premises have been presented again. Directly stating those conclusions provides a benefit that repeating the premises does not reproduce.

The same retained information can support different outcomes depending on how it is processed before action (Figure 3C). Guided Preparation completes all 60 tasks, compared with 21 under direct execution. Mandatory Reconstruction completes 42, improving on D but remaining below Guided

Table 1: TaskPass across benchmarks and executors.

Benchmark	Model	*Explicit*	D / P0	Δ(pp)
TextWorld	Qwen BF16	58/60 (96.7%)	21/60 (35.0%)	+61.7
	Qwen BF16	45/60 (75.0%)	17/60 (28.3%)[P0]	+46.7
	Qwen BF16, Thinking	60/60 (100%)	60/60 (100%)[P0]	0.0
BabyAI	Qwen FP8, Thinking, 8K	50/60 (83.3%)	41/60 (68.3%)	+15.0
	Qwen FP8, Thinking, 16K	55/60 (91.7%)	55/60 (91.7%)	0.0
	DeepSeek-V4-Flash	57/60 (95.0%)	60/60 (100%)	-5.0
	GLM-5.3-Flash	53/60 (88.3%)	58/60 (96.7%)	-8.3
ALFWorld	Qwen BF16	16/100 (16.0%)	1/100 (1.0%)	+15.0
	DeepSeek-V4-Flash	33/100 (33.0%)	39/100 (39.0%)	-6.0

[P0] Marked entries use neutral padding (P0) from Figure 3B, and other entries use Derivable (D). Differences are Explicit minus the indicated comparison, in percentage points. All BabyAI rows use the same 60 task identities. The Qwen 8K row is taken from the dual-budget execution so that it is paired with the 16K row. A separate Qwen 8K execution on the same task identities gives 52/60 under Explicit and 36/60 under Derivable in Appendix C.7. The first TextWorld row uses the preparation sample. The two P0-marked rows use the same separate set of 60 tasks with thinking disabled and enabled.



(A) Derived information.

(B) Premise repetition.

(C) Preparation before action.

(D) Generation budget in BabyAI.

Figure 3: Content, preparation, and generation-budget effects on TaskPass. Each panel uses a separate 60-task sample, so comparisons across panels are not paired. In A, points average two repeats per task and error bars show 95% task-paired bootstrap intervals. B–D show observed success rates.

Preparation and Explicit. Across all task attempts, C and R use an average of 58.2K and 56.9K total tokens, respectively, compared with 35.2K for E (Appendix Table 12). These totals include preparation and action calls and vary with episode length. Both preparation conditions add a call before action and share the Derivable input, a public reconstruction procedure, and the preparation budget. In a subsequent independent execution on the same tasks, adding an adapted description of this procedure to D without a preparation call gives 20/60 successes (Appendix C.6). This description

Table 2: Same-state action effects across environments and executors.

Benchmark	Model / comparison	N	Direct rescue	Reverse	Δ (pp)	95% CI	Exact p
TextWorld	Qwen BF16, E–D	60	31	0	+51.7	[38.3,65.0]	9.31×10^{-10}
	Qwen BF16, E–P_REPEAT	60	39	0	+65.0	[53.3,76.7]	3.64×10^{-12}
ALFWorld	Qwen BF16, E–D	100	9	1	+8.0	[2.0,14.0]	.0215
	Gemma 4 31B IT, E–D	100	11	1	+10.0	[4.0,17.0]	.00635
	DeepSeek-V4-Flash, E–D	100	8	7	+1.0	[-7.0,9.0]	1.000

Direct rescue counts pairs in which only the action selected under E is applicable, and Reverse counts the opposite case. Δ is 100 times the difference between these counts divided by N. Qwen and Gemma use thinking disabled. Intervals are task-paired bootstrap 95% intervals, and exact p values are uncorrected two-sided sign tests.

alone does not reproduce the preparation gain. C allows free-form notes, while R requires a complete registry. Their difference shows that full reconstruction is not the only effective way to prepare from the retained information.

The BabyAI comparison asks whether allowing more computation before action also changes the difference between Explicit and Derivable (Figure 3D). On the same 60 tasks, increasing the generation limit from 8,192 to 16,384 tokens improves both conditions and brings their observed TaskPass to the same level. At the lower limit, 19 Derivable executions and nine Explicit executions are truncated before returning an action, and fewer executions are truncated at the higher limit. Task completion therefore depends partly on whether generation reaches an action. The E–D difference is 15 percentage points at 8K (95% CI [0,30]) and zero at 16K (95% CI [-10,10]). The interval for their change also includes zero. Additional budget comparisons appear in Appendix C.7. Table 2 complements these task-level results by comparing actions from identical environment states.

5.3 Same-State Analysis

Table 2 compares actions selected from a shared environment state. In TextWorld, temperature zero and deterministic transitions mean that task-level divergence necessarily begins at a shared state. The informative quantity is therefore the direction of the first discordant decision, since the existence of a shared-state divergence follows from the design. Explicit is more likely to select an action whose preconditions are satisfied. This occurs against both Derivable and premise repetition, with no reverse case in either comparison.

The comparison in ALFWorld uses the same 100 initial states for Qwen, Gemma, and DeepSeek, selected to include both applicable and inapplicable operations (Table 2). With thinking disabled, Explicit improves action applicability by 10 percentage points for Gemma and 8 points for Qwen, while DeepSeek is nearly balanced. Positive action effects thus extend to a second environment and another model family. The observed gains differ across these executor configurations. Qwen with thinking enabled has an observed difference of +4 points, but many outputs end before an action is returned. The exact paired test and bounds for the unreturned actions leave this result uncertain (Appendix C.9).

Table 10 in Appendix C illustrates why action applicability and task success must be considered separately. In one case, Explicit selects an applicable preparation operation while P_REPEAT repeats a completed inspection, yet both runs eventually fail. In another, both actions are applicable, but P_REPEAT selects an operation that causes irreversible failure while Explicit selects unlock and later succeeds. The interface can affect both whether the next action can be executed and which executable action is chosen.

6 Discussion and Implications

Exact derivability does not establish behavioral equivalence. Exact derivability is a useful interface property, but behavioral preservation must be evaluated separately for a learned executor. The present evidence concerns derived state in the current-state interface and does not test the compression

of interaction history. ALFWorld includes task-level comparisons for Qwen and DeepSeek and fixed-state comparisons for Qwen, Gemma, and DeepSeek.

Directional action effects appear from shared states. In TextWorld, deterministic transitions at temperature zero make a shared-state origin necessary for any first task-level divergence, so the location is not an independent mechanism finding. The substantive result is directional. All 31 differences in action applicability at the first E–D divergence favor Explicit. ALFWorld provides a complementary comparison on pre-selected shared states that are not reached through earlier model choices.

Full reconstruction is not required for effective preparation. The direct-action procedure control remains close to the original D result, while C and R achieve higher task success. C and R share the original procedure and the same preparation budget. Their difference shows that requiring a complete registry can be less effective than allowing the model to prepare through free-form notes.

The effect varies with pre-action computation. Across the Qwen task-completion comparisons, the observed difference appears with thinking disabled or at the lower generation limit and is absent in some settings with thinking or a higher generation limit. This pattern is consistent with sensitivity to computation available before action. All 60 TextWorld preparation tasks have the same maximum derivation depth, so this sample cannot identify task-level depth effects (Appendix C.3). With thinking enabled, all three conditions in the matched TextWorld control succeed on every task, while Explicit uses more completion tokens (Appendix C.5). Sufficient model-side computation can therefore remove the success-rate difference, although stating the derived state does not reduce completion use in this setting.

7 Conclusion

This paper shows that derivability alone does not guarantee that omitting information preserves LLM agent behavior. Directional action-applicability effects also appear from shared states in TextWorld and ALFWorld. Thinking, a higher generation limit, or free-form preparation before action removes or reduces the observed difference in some settings, and the effect also varies across executors. Behavioral claims about derivable omission should therefore state the executor, inference settings, and pre-action processing under which they were evaluated.

AI use statement

Generative AI tools, including Anthropic Claude, OpenAI ChatGPT and Codex, were used as research assistants throughout this project. They assisted with conceptual framing and hypothesis refinement, experimental design and protocol development, implementation and review of research code and analyses, literature search and summarization, result interpretation, translation, and manuscript drafting and editing. AI-assisted outputs were reviewed by the authors and checked against experimental records, analysis results, or source materials as appropriate. Generative AI did not serve as the ground-truth evaluator for TaskPass or exact reconstruction, nor did it determine statistical significance. The authors take responsibility for the final content, claims, and artifacts in this paper.

Ethics statement

This work studies LLM agents in controlled benchmark and simulator environments. It does not involve human-subject experimentation, collection of private user data, or deployment in safety-critical real-world settings. The findings characterize controlled agent-interface interventions under the evaluated tasks and model configurations and should not be read as direct measurements of real-world deployment behavior.

Reproducibility statement

The main text describes the interface intervention, task populations, model configurations, paired execution design, outcome measures, and statistical analyses. Appendix A documents exact reconstruction, environment-specific semantics, worked examples, and construct-validation checks.

Appendix B provides task and regime-selection details, model and runtime settings, execution-integrity information, statistical procedures, and the termination and uncertainty analyses underlying the reported results. The anonymous supplementary reproduction package contains experiment code, archived task and outcome records, and scripts for offline reconstruction and statistical analysis. Its documentation identifies the available evidence and requirements for new model executions.

REFERENCES

Zhaoyu Bai and Jiaqi Cai. PatchOptic for Shared-State LLM Workflows with Projected Views and Verified Structured Updates, 2026. URL https://arxiv.org/abs/2607.05483.

Albert Benveniste, Benoît Caillaud, Dejan Nickovic, Roberto Passerone, Jean-Baptiste Raclet, Philipp Reinkemeier, Alberto L. Sangiovanni-Vincentelli, Werner Damm, Thomas A. Henzinger, and Kim G. Larsen. Contracts for System Design. *Foundations and Trends in Electronic Design Automation*, 12(2–3):124–400, 2018. doi: 10.1561/1000000053. URL https://doi.org/10.1561/1000000053.

Lukas Berglund, Meg Tong, Maximilian Kaufmann, Mikita Balesni, Asa Stickland, Tomek Korbak, and Owain Evans. The Reversal Curse: LLMs trained on "A is B" fail to learn "B is A". In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/5178b2f2d7c44aa390c0777dc77b3f0c-Abstract-Conference.html.

Yuanqing Cai, Ziyi Huang, Minhao Liu, Lixin Duan, Wen Li, and Yanru Zhang. MixRea: Benchmarking Explicit-Implicit Reasoning in Large Language Models. arXiv, 2026. URL https://arxiv.org/abs/2605.20128.

Maxime Chevalier-Boisvert, Dzmitry Bahdanau, Salem Lahlou, Lucas Willems, Chitwan Saharia, Thien Huu Nguyen, and Yoshua Bengio. BabyAI: A Platform to Study the Sample Efficiency of Grounded Language Learning. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rJeXCo0cYX.

Peter Clark, Oyvind Tafjord, and Kyle Richardson. Transformers as Soft Reasoners over Language. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pp. 3882–3890, 2020. doi: 10.24963/ijcai.2020/537. URL https://www.ijcai.org/Proceedings/2020/537.

Marc-Alexandre Côté, Ákos Kádár, Xingdi Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, James Moore, Matthew Hausknecht, Layla El Asri, Mahmoud Adada, Wendy Tay, and Adam Trischler. TextWorld: A Learning Environment for Text-Based Games. In *Computer Games*, volume 1017 of *Communications in Computer and Information Science*, pp. 41–75. Springer, 2019. doi: 10.1007/978-3-030-24337-1_3. URL https://link.springer.com/chapter/10.1007/978-3-030-24337-1_3.

I. de Zarzà, J. de Curtò, Jordi Cabot, Pietro Manzoni, and Carlos T. Calafate. Semantic Invariance in Agentic AI. arXiv, 2026. URL https://arxiv.org/abs/2603.13173.

Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jiang, Bill Yuchen Lin, Sean Welleck, Peter West, Chandra Bhagavatula, Ronan Le Bras, Jena Hwang, Soumya Sanyal, Xiang Ren, Allyson Ettinger, Zaid Harchaoui, and Yejin Choi. Faith and Fate: Limits of Transformers on Compositionality. In *Advances in Neural Information Processing Systems*, volume 36, 2023. doi: 10.52202/075280-3081. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/deb3c28192f979302c157cb653c15e90-Abstract.html.

Guhao Feng, Bohang Zhang, Yuntian Gu, Haotian Ye, Di He, and Liwei Wang. Towards Revealing the Mystery behind Chain of Thought: A Theoretical Perspective. In *Advances in Neural Information Processing Systems*, volume 36, 2023. doi: 10.52202/075280-3100. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/dfc310e81992d2e4cedc09ac47eff13e-Abstract-Conference.html.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. PAL: Program-aided Language Models. In *International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 10764–10799, 2023. URL https://proceedings.mlr.press/v202/gao23f.html.

Xinyu Guan, Qianyang Zhao, and Yuming Deng. Decision-Aware Memory Cards: Counterfactual-Inspired Context Selection and Compression for Tool-Using LLM Agents, 2026. URL https://arxiv.org/abs/2606.08151.

Huiqiang Jiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. LLMLingua: Compressing Prompts for Accelerated Inference of Large Language Models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 13358–13376. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.825. URL https://aclanthology.org/2023.emnlp-main.825/.

Minki Kang, Wei-Ning Chen, Dongge Han, Huseyin A. Inan, Lukas Wutschitz, Yanzhi Chen, Robert Sim, and Saravan Rajmohan. ACON: Optimizing Context Compression for Long-horizon LLM Agents. In *International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2510.00615.

Imene Kerboua, Sahar Omidi Shayegan, Megh Thakkar, Xing Han Lù, Léo Boisvert, Massimo Caccia, Jérémy Espinas, Alexandre Aussem, Véronique Eglin, and Alexandre Lacoste. FocusAgent: Simple Yet Effective Ways of Trimming the Large Context of Web Agents. *Transactions on Machine Learning Research*, 2026. URL https://openreview.net/forum?id=mINaJKSy7A.

Philippe Laban, Hiroaki Hayashi, Yingbo Zhou, and Jennifer Neville. LLMs Get Lost In Multi-Turn Conversation. In *ICLR 2026*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/59f6421e64707225fdf5b28840679a07-Abstract-Conference.html.

Zhiyuan Li, Hong Liu, Denny Zhou, and Tengyu Ma. Chain of Thought Empowers Transformers to Solve Inherently Serial Problems. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/3309b4112c9f04a993f2bbdd0274bba1-Abstract-Conference.html.

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. URL https://aclanthology.org/2024.tacl-1.9/.

Alejandro J. Mercado and Alessio Lomuscio. Formal Verification of Agentic Systems over Operational Data, 2026. URL https://arxiv.org/abs/2608.03609.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah Smith, and Mike Lewis. Measuring and Narrowing the Compositionality Gap in Language Models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 5687–5711, 2023. doi: 10.18653/v1/2023.findings-emnlp.378. URL https://aclanthology.org/2023.findings-emnlp.378/.

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. URL https://openreview.net/forum?id=0IOX0YcCdTn.

Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. ProofWriter: Generating Implications, Proofs, and Abductive Statements over Natural Language. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 3621–3634, 2021. doi: 10.18653/v1/2021.findings-acl.317. URL https://aclanthology.org/2021.findings-acl.317/.

Sylvie Thiébaux, Jörg Hoffmann, and Bernhard Nebel. In Defense of PDDL Axioms. *Artificial Intelligence*, 168(1–2):38–69, 2005. doi: 10.1016/j.artint.2005.05.004. URL https://doi.org/10.1016/j.artint.2005.05.004.

Haoyu Wang, Christopher M. Poskitt, and Jun Sun. AgentSpec: Customizable Runtime Enforcement for Safe and Reliable LLM Agents. In *Proceedings of the 48th IEEE/ACM International Conference on Software Engineering*, pp. 2938–2950, 2026a. doi: 10.1145/3744916.3764546. URL https://conf.researchr.org/details/icse-2026/icse-2026-research-track/29/AgentSpec-Customizable-Runtime-Enforcement-for-Safe-and-Reliable-LLM-Agents.

Jincheng Wang, Min Zheng, and Tao Wei. COVENANT: Natural-Language Workflow Compilation for Aligned Agent Execution, 2026b. URL https://arxiv.org/abs/2607.25400.

Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. Plan-and-Solve Prompting: Improving Zero-Shot Chain-of-Thought Reasoning by Large Language Models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2609–2634, 2023. doi: 10.18653/v1/2023.acl-long.147. URL https://aclanthology.org/2023.acl-long.147/.

Yilun Xu, Shengjia Zhao, Jiaming Song, Russell Stewart, and Stefano Ermon. A Theory of Usable Information under Computational Constraints. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=r1eBeyHFDH.

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. URL https://arxiv.org/abs/2210.03629.

Tianyun Zhong, Wangyi Jiang, Wei Wang, Xuanang Chen, Yaojie Lu, Shiwei Ye, Yuzhen Shi, Boyu Yang, Jinghang Wang, Han Li, Weiqi Zhai, Bing Zhao, Hu Wei, Haiyang Yu, Yongbin Li, Hongyu Lin, Le Sun, and Xianpei Han. Before the Action: Benchmarking LLMs on Prospective Hypothesis Discovery. arXiv, 2026. URL https://arxiv.org/abs/2607.15766.

A Interface Construction and Validation

A.1 Reconstruction contract and formal definition

A task is represented as a finite-horizon state-transition system $\mathcal{M} = (\mathcal{S}, \mathcal{A}, T, \mathrm{obs}, \mathcal{G})$, where $\mathcal{S}$ is the environment state space, $\mathcal{A}$ is the structured action space, T is the transition function, obs is the observation mapping, and $\mathcal{G}$ is the task-success criterion. The paired interfaces share the task, action space, transition dynamics, observation mechanism, and evaluator. They differ only in whether selected task-relevant derived state is rendered directly to the model.

At a state s, the *Derivable* interface retains the agent-visible facts and public rules used to reconstruct the omitted derived state under the declared semantics. The *Explicit* interface contains the same common payload and additionally renders that derived state. Qualification is statewise. For every checker-reachable state within the declared horizon, the deterministic reconstruction g_Σ applied to the *Derivable* interface must reproduce the canonical *Explicit* interface exactly, as required by Eq. 2.

The reconstruction uses only the retained agent-visible interface and the declared public semantics. Hidden simulator state, evaluator-only information, plans, and action hints are not available to the reconstruction function.

A.2 Environment-specific public semantics

The reconstruction criterion is shared across environments, but the public semantics are environment-specific.

TextWorld. The derived-state rules are positive conjunctive implications with a single conclusion. Starting from currently true entries in the agent-visible Boolean base-fact map, the checker repeatedly adds any conclusion whose premises are satisfied until reaching the least fixed point. A target rendered as true must have a positive derivation in this closure. A target rendered as false denotes non-membership in the positive closure under this formalism. It is not interpreted as a proof of logical negation.

BabyAI. BabyAI uses a separate total Boolean registry whose features are computed from the public agent-visible state. A false value is explicitly defined by this registry's Boolean semantics. The two environments use different public semantics but the same qualification principle. The complete *Explicit* interface must be exactly reconstructible from the *Derivable* interface and the declared public semantics.

ALFWorld. ALFWorld derives goal-relevant object and receptacle properties from the retained PDDL goal and base facts. The public rules identify type matches, containment and location, container access, whether an object is held or colocated with the agent, and the status of goal-required properties. Explicit adds the resulting conclusions as supplemental context. Derivable retains the same goal, base facts, object types, action schemas, and derivation rules. These conclusions summarize the current state and do not supply a plan or an admissible-action list.

A.3 MODEL-VISIBLE INTERFACES

TextWorld request. The direct-action request is stateless. It consists of one system message and one freshly serialized current-state user payload, followed by the tool definition. Previous turns are not accumulated. The system message reads.

> You operate a deterministic symbolic environment. Use the execute_operation tool exactly once per turn. Choose an operation whose frozen public preconditions hold and whose public effects advance the stated goal assertion. Do not emit prose and do not invent operation identifiers.

The user payload contains a versioned task contract, public Boolean base facts, ordered public rules, the goal assertion, remaining action budget, and the catalog of opaque operation identifiers. The catalog describes public preconditions and effects through `requires_true`, `sets_true`, `sets_false`, and `public_observation`. The task-specific tool schema has one required `operation_id`, enumerates its admissible identifiers, and disallows additional properties. This is the syntactic action catalog and does not list the actions currently applicable at the given state. *Explicit* adds the exact current fixed-point registry, which *Derivable* omits. In the BF16 preparation comparison, it appears in the top-level `derived_registry` field. The same rules and declared action semantics remain visible in both arms. The catalog groups branch-specific transitions under public operation identifiers and lists conditions on public base predicates and their public effects. In the 60-task BF16 preparation comparison, no public operation lists a derived predicate as a precondition.

BabyAI request. The BabyAI interface likewise uses one system message, one current public-state message, and the `execute_action` tool. The system message reads.

> Operate the partially observed grid world and satisfy the mission. Use execute_action exactly once. Directions are 0=east, 1=south, 2=west, 3=north. forward moves into an empty/open-door cell; pickup and toggle act on the cell ahead. Use the current observation, memory, rules, and explicitly materialized derived assertions.

The current payload contains partial observation, public memory, mission-verifier state, clock, ordered registry rules, and remaining budget. The tool takes an object with the sole key `action`, with values `left`, `right`, `forward`, `pickup`, `drop`, `toggle`, or `done`. *Explicit* adds the Boolean `derived` registry, while *Derivable* omits it. Both conditions receive the same system message, including its reference to derived assertions. This wording is also present in D even though D omits the `derived` field, so it could disadvantage D at the lower budget. The TextWorld and ALFWorld prompts do not contain the same reference. In BabyAI, D matches E at 16K and matches or exceeds E for DeepSeek and GLM despite the shared wording.

ALFWorld request. The current-state request contains the PDDL goal, typed objects, base facts, action schemas, public derivation rules, and remaining action budget. Both conditions use the same opaque object aliases and schema-specific tools. Explicit adds the derived conclusions in `supplemental_context`. The same-state comparisons request one action from each shared initial state.

Preparation and repetition controls. P_REPEAT repeats retained premises and rules without inserting derived conclusions. In the preparation control, C and R receive the same public least-fixed-point reconstruction procedure and preparation budget. That procedural text is not included in the D or E action requests. Comparisons of C or R with D therefore change both pre-action processing and procedure wording. C allows free-form notes, while R requests a complete registry. Both preparation prompts prohibit plans and action recommendations, and neither includes the operation catalog. The subsequent action request receives the model's preparation output. E receives the system-computed derived registry directly.

At each decision, C and R receive the same Derivable interface, public least-fixed-point procedure, registry order, and remaining action budget. Each preparation call and each subsequent action call has a 4,096-token generation limit, with temperature zero and thinking disabled. C returns a string of free-form notes through `submit_preparation`. R returns every registry predicate through `submit_derived_registry`, using TRUE, FALSE, or UNKNOWN. The action request receives C's notes or R's generated registry without correction. The correct registry supplied to E is computed by the deterministic rule evaluator.

A.4 Worked interface examples

BabyAI. The worked example uses the mission "open the blue door, then go to the green ball." At the initial state, the visible memory contains a locked blue door at $(4, 3)$ and a blue key at $(2, 4)$. The public registry maps these observations to `matching_door_observed=true` and `matching_key_observed=true`, which support the rule

$$\texttt{matching_door_observed} \wedge \texttt{matching_key_observed} \Rightarrow \texttt{barrier_identity_resolved}. \tag{4}$$

The common payload is identical across the paired interfaces. *Explicit* adds the complete 12-entry `derived` object, including `barrier_identity_resolved=true`, while *Derivable* omits that object.

TextWorld. A complementary example illustrates both multi-hop derivation and state-dependent recomputation. In one reachable state, $\texttt{v_0} = \text{true}$ and $\texttt{ready_1} = \text{true}$, together with the rules

$$\texttt{v_0} \rightarrow \texttt{d_0}, \qquad \texttt{d_0} \wedge \texttt{ready_1} \rightarrow \texttt{d_2}, \tag{5}$$

imply $\texttt{d_0} = \text{true}$ and $\texttt{d_2} = \text{true}$. A subsequent legal operation changes `pending_0` from true to false and `ready_0` from false to true. Recomputing the public closure then adds `d_1` and `d_3`. This example shows why reconstruction is checked again after state transitions.

A.5 Construct validation

The interface checker was evaluated on a separate 30-task validation set before the behavioral experiments. Table 3 summarizes the coverage of this test suite.

Table 3: Construct-validation summary for the interface checker.

Validation item	Result
Tasks passing all checks	30/30
Checker-reachable states	1,020
Post-transition witnesses	1,260
Minimum omitted target assertions per task	≥ 8
Observation-dependent branch	30/30
Irreversible action / dynamic recomputation	30/30 / 30/30
Forbidden plan/action strings detected	0/30
Seeded mutations detected	11,565/11,565

The suite tests reconstruction, state transitions, and dynamic recomputation independently of the behavioral experiments. The string check searches the lowercase JSON representation of each rendered interface for `winning_policy`, `policy_commands`, `walkthrough`, `admissible_commands`, `complete_plan`, `next_step`, `expert_plan`, `golden`, and `solution`.

B Experimental and Statistical Details

B.1 Benchmarks and task populations

TextWorld. Experiments use TextWorld 1.7.0. An *Explicit*-only calibration over low, medium, and high difficulty selected the medium regime. A separate FP8 population contains 60 medium tasks, split evenly between structural profiles $(3, 1)$ and $(3, 2)$, with condition order balanced within each profile. Both profiles have three preparation operations and two branches, with one and two distractor operations, respectively. The BF16 E/D comparison uses a separate sample of 60 tasks. The two populations are analyzed separately. A separate 30-task population is used for construct validation.

The Explicit-only calibration used ten tasks at each of the low, medium, and high difficulty levels. TaskPass was 9/10, 8/10, and 10/10, respectively. The prespecified selection rule required a TaskPass between 40% and 80%, inclusive. Medium was the only eligible level and was selected without observing Derivable outcomes. The following comparisons each use a separate 60-task sample.

Comparison	Conditions and purpose
BF16 preparation	D, E, C, and R in Table 1 and Figure 3C. The direct-action procedure control reuses these 60 tasks in a subsequent execution.
Derived content	D, P0, P50, and E under BF16 and FP8 in Figure 3A.
Premise repetition	P0, P_REPEAT, and E with thinking disabled and enabled in Figure 3B.
Same-state repetition	E and P_REPEAT for action comparison in Table 2.
Termination	E and P_REPEAT under early termination and continuation after rejected actions.
FP8 E/D	Explicit and Derivable with results by structural profile in Table 5.

Explicit success ranges from 75.0% to 96.7% across the thinking-disabled samples and reaches 100% in the thinking-enabled comparison. Each sample is analyzed separately.

BabyAI. Experiments use BabyAI/MiniGrid v3.1.0. The cross-executor comparison uses 60 tasks with 30 near-east and 30 near-south layouts and a 12-step horizon. Qwen, DeepSeek, and GLM share these task identities. The E/D budget comparison in Figure 3D evaluates the same tasks in separate executions at 8K and 16K, giving 240 task-condition-budget observations. A three-condition budget comparison includes P_REPEAT and uses 60 different tasks with no overlap. Appendix C.7 reports each comparison separately.

ALFWorld. Experiments use official PDDL tasks from the `valid_unseen` split. Tasks were eligible when the state used for action comparison had at least one applicable and one inapplicable operation and passed the reconstruction and structural checks. A fixed hash ordering selected 100 of 147 eligible tasks. This sample is used for Qwen and DeepSeek task completion and the Qwen, DeepSeek, and Gemma comparisons on shared initial states. Qwen thinking-OFF and thinking-ON are evaluated on the same states.

B.2 Models and inference configurations

The settings in Table 4 apply to the main comparisons.

B.3 Actions, evaluation, and termination

TextWorld. Exactly one `execute_operation` call must contain a task-enumerated opaque operation identifier. Missing, multiple, malformed, or unknown calls are parser-invalid. The checker

Table 4: Model and inference settings used by the main experiments.

Study	Model and inference setting
TextWorld BF16 E/D	Qwen3.8-27B BF16, thinking OFF, temperature 0, maximum generation 4096, $N = 60$.
TextWorld dose / repetition / preparation	Qwen3.8-27B BF16 or FP8, thinking OFF, temperature 0, maximum completion 4096.
TextWorld repetition, thinking ON	Qwen3.8-27B BF16, temperature 0, maximum completion 4096, $N = 60$.
BabyAI Qwen dual budget	Qwen3.8-27B FP8, thinking ON, temperature 0, maximum completion 8192 or 16384 on the same frozen $N = 60$.
BabyAI DeepSeek	DeepSeek-V4-Flash, temperature 0, top-p .95, maximum completion 4096.
BabyAI GLM	GLM-5.3-Flash, temperature 0, top-p .95, maximum completion 65536. The thinking setting was not recorded.
ALFWorld DeepSeek	DeepSeek-V4-Flash, temperature 0, top-p .95, generation limit 4096, action budget 18.
ALFWorld Qwen task completion	Qwen3.8-27B BF16, thinking OFF, temperature 0, top-p .95, generation limit 4096, action budget 18, $N = 100$.
ALFWorld Qwen same-state	Qwen3.8-27B BF16, temperature 0, top-p .95, generation limit 4096 with thinking OFF or 16384 with thinking ON, 100 shared initial states.
ALFWorld Gemma	Gemma 4 31B IT BF16, temperature 0, thinking OFF, generation limit 512, 100 shared initial states.

resolves a valid identifier before testing current preconditions. An inapplicable action is distinct from a parsing failure. Under early termination, an episode ends at exact goal satisfaction, parser invalidity, first inapplicability, or the seven-action horizon. Under continuation after rejected actions, rejection leaves the world state unchanged, consumes one step, and returns only `Rejected.` before execution continues.

BabyAI. The sole argument is one of the seven action names. Official deterministic action semantics determine whether forward movement or manipulation changes the world. An unmet precondition produces a no-op that still advances the public tick. It is not itself a parser error. TaskPass requires `verifier_status == success`. Terminals distinguish verifier success, environment/verifier failure, parser invalidity, and horizon exhaustion. The low, medium, and high difficulty levels allow 12, 18, and 24 steps, respectively. The shared 60-task low-difficulty sample uses 12. The native-thinking comparison additionally reports truncation separately, including cases where truncation prevents a valid tool action.

ALFWorld. Exactly one `pddl_<ACTION>` call must name an existing PDDL schema. Argument keys must equal its parameter keys and all opaque object aliases must resolve. The PDDL checker determines current applicability and exact goal truth. During task execution, inapplicability does not change world state, consumes one action, and returns `action_not_applicable`. Goal success, parser invalidity, and the 18-action budget terminate an episode. Same-state comparisons evaluate one action from each shared initial state. Applicability is benchmark-specific and is not pooled across environments.

B.4 Primary paired analysis

The independent analysis unit is the task. For each task, the *Explicit* and *Derivable* TaskPass outcomes form one paired binary observation. The primary estimand is

$$\Delta = \Pr(\text{pass} \mid \text{Explicit}) - \Pr(\text{pass} \mid \text{Derivable}). \tag{6}$$

Uncertainty is estimated by 100,000 deterministic task-paired bootstrap resamples, and the exact two-sided McNemar test is computed from the discordant pairs.

The BF16 TextWorld comparison uses 100,000 task-paired bootstrap resamples with seed 130920261. The three primary comparisons are E–D, R–C, and E–R, with Holm correction across that family.

A separate FP8 60-task TextWorld comparison passed 47/60 tasks (78.3%) under *Explicit* and 25/60 (41.7%) under *Derivable*, for +36.7 percentage points. Its pooled pair table is $n_{10}/n_{11}/n_{01}/n_{00} = 22/25/0/13$. The two predeclared medium structural profiles are reported below. This FP8 population is analyzed separately from the BF16 population in Table 1. Complete serving metadata is unavailable for this FP8 comparison.

Table 5: FP8 profile-specific paired results. Effects are percentage points, intervals are 100,000-resample task-paired bootstrap 95% intervals.

Profile	**Explicit**	**Derivable**	**Δ**	**Turnover**	**McNemar p**
$(3, 1)$	76.67%	30.00%	+46.67	46.67%	0.0001221
$(3, 2)$	80.00%	53.33%	+26.67	26.67%	0.0078125

The corresponding bootstrap intervals for Δ are $[30.00, 63.33]$ pp for $(3, 1)$ and $[13.33, 43.33]$ pp for $(3, 2)$.

B.5 Fixed-slot dose analysis

The fixed-slot dose study crosses 60 tasks with two Qwen precision configurations, four interface conditions, and two repeated executions, yielding 960 behavioral cells. Repeats are nested within task and are never treated as independent tasks. Within each precision and condition, the two executions agree on TaskPass for 59 or 60 of the 60 tasks. The four conditions are D, P0, P50, and E. D omits the supplemental slot, P0 provides the matched scaffold without target derived entries, P50 exposes half of the target derived entries, and E exposes the complete target derived state.

The analysis uses Python 3.12.14 and NumPy 2.3.5. Its pre-specified PCG64 seed is `91740235027927233`. Bootstrap resampling is performed at the task level within the two structural profiles while preserving checkpoint, condition, and repeat structure. The analysis uses 100,000 bootstrap replicates.

The primary hypothesis family contains nine two-sided exact conditional sign-flip tests. P0–D, P50–P0, E–P50, and E–P0 within each checkpoint, plus the BF16-versus-FP8 interaction in E–P0. Holm correction controls the familywise error rate at $\alpha = .05$. The predeclared equivalence band is ± 10 percentage points. A nonsignificant difference is not interpreted as equivalence. Table 6 reports the complete primary family.

Table 6: Complete primary test family for the fixed-slot dose analysis. E provides the complete selected derived state.

Estimand	**Effect (pp)**	**95% CI (pp)**	**Exact p**	**Holm p**
BF16 E–P0	+50.00	$[38.33, 63.33]$	1.86×10^{-9}	1.49×10^{-8}
BF16 P0–D	+5.83	$[-5.00, 16.67]$	0.3877	0.7754
BF16 P50–P0	+25.83	$[11.67, 40.00]$	0.001544	0.006176
BF16 E–P50	+24.17	$[12.50, 35.83]$	0.0002747	0.001648
FP8 E–P0	+55.00	$[41.67, 66.67]$	2.33×10^{-10}	2.10×10^{-9}
FP8 P0–D	+10.83	$[-2.50, 24.17]$	0.1435	0.4304
FP8 P50–P0	+30.00	$[16.67, 43.33]$	0.0002772	0.001648
FP8 E–P50	+25.00	$[15.00, 36.67]$	6.10×10^{-5}	0.0004272
BF16–FP8 interaction	-5.00	$[-15.00, 5.00]$	0.5078	0.7754

Mean first-prompt token counts decrease from 5,134.97 under P0 to 5,103.13 under P50 and 5,071.03 under E while TaskPass increases over the same sequence. This ordering is inconsistent with a simple

monotonic account in which more prompt tokens produce better performance. It does not isolate a pure content effect or rule out composition, position, or tokenizer-level differences.

Whole-episode token use follows a different pattern. From the recorded task-level usage, mean total input and output tokens per attempt for P0, P50, and E are 25,952, 31,231, and 35,391 in BF16, and 25,591, 31,446, and 35,746 in FP8. These means include failures and average the two executions of each of the 60 tasks. Mean calls increase from 5.03 to 6.07 to 6.93 in BF16 and from 4.97 to 6.12 to 7.00 in FP8. Thus shorter initial prompts coexist with greater cumulative use as executions continue for more steps.

C Additional Results and Behavioral Diagnostics

C.1 Results summary

Table 7 collects the experiments used in the paper's main claims and scope analysis.

Table 7: Summary of the main endpoint and same-state results. Positive differences favor direct exposure of derived state.

Study	Comparison	Main result
TextWorld BF16, $N = 60$	Explicit vs. Derivable	58/60 vs. 21/60, $\Delta = +61.7$ pp
TextWorld dose, BF16	E vs. P0	$\Delta = +50.0$ pp, 95% CI $[38.33, 63.33]$
TextWorld dose, FP8	E vs. P0	$\Delta = +55.0$ pp, 95% CI $[41.67, 66.67]$
TextWorld matched repetition, $N = 60$	P0 / P_REPEAT / E	17/60, 12/60, 45/60 TaskPass
TextWorld matched repetition, thinking ON, $N = 60$	P0 / P_REPEAT / E	60/60, 60/60, 60/60 TaskPass
TextWorld termination control, first inapplicable action, $N = 60$	E vs. P_REPEAT	54/60 vs. 15/60, $\Delta = +65.0$ pp, 95% CI $[51.67, 76.67]$
TextWorld termination control, continue after rejection, $N = 60$	E vs. P_REPEAT	53/60 vs. 14/60, $\Delta = +65.0$ pp, 95% CI $[50.00, 78.33]$
TextWorld preparation, $N = 60$	D / C / R / E	21/60, 60/60, 42/60, 58/60
BabyAI Qwen FP8 thinking ON, final 8K, $N = 60$	Explicit vs. Derivable	50/60 vs. 41/60, $\Delta = +15.0$ pp, exact $p = 0.093$
BabyAI Qwen FP8 thinking ON, final 16K, $N = 60$	Explicit vs. Derivable	55/60 vs. 55/60, $\Delta = 0$ pp, exact $p = 1.0$
BabyAI DeepSeek, $N = 60$	Explicit vs. Derivable	57/60 vs. 60/60, $\Delta = -5.0$ pp
BabyAI GLM, $N = 60$	Explicit vs. Derivable	53/60 vs. 58/60, $\Delta = -8.33$ pp
ALFWorld Qwen thinking OFF, $N = 100$	Explicit vs. Derivable	16/100 vs. 1/100, $\Delta = +15.0$ pp, 95% CI $[8.0, 23.0]$
ALFWorld DeepSeek, $N = 100$	Explicit vs. Derivable	33/100 vs. 39/100, $\Delta = -6.0$ pp
ALFWorld Qwen thinking OFF, $N = 100$	same-state action	9 direct rescues, 1 reverse, $+8$ pp, 95% CI $[+2, +14]$
ALFWorld Qwen thinking ON, $N = 100$	same-state action	Observed-output difference $+4$ pp, exact $p = .125$. Truncation bounds appear in Appendix C.9
ALFWorld Gemma 4, $N = 100$	same-state action	11 direct rescues, 1 reverse, $+10$ pp, 95% CI $[+4, +17]$
ALFWorld DeepSeek, $N = 100$	same-state action	8 direct rescues, 7 reverse, $+1$ pp, 95% CI $[-7, +9]$

On the tasks of Figure 3B, Explicit exceeds P0 by 46.67 percentage points (95% CI [31.67, 60.00]) and P_REPEAT by 55.00 points (95% CI [41.67, 68.33]) with thinking disabled. With thinking enabled, every condition completes all 60 tasks. In the termination comparison, the E-P_REPEAT difference is the same under both rules, and the paired difference between the two rules is 0.0 points (95% CI [-6.67, +6.67]).

C.2 Controls for the content effect

Table 8 summarizes the main controls used to distinguish derived-content exposure from simpler presentation effects.

Table 8: Controls for the derived-content effect.

Question	Control	Result	Interpretation
Does any extra block help?	P0 vs. D	Small differences in BF16 and FP8	The scaffold alone does not reproduce E.
Does derived-content amount matter?	P0 → P50 → E	TaskPass rises in both checkpoints	Performance increases with stated derived content.
Is repeating retained evidence sufficient?	P_REPEAT vs. E	E gains 55 pp in the matched comparison	Premise repetition does not reproduce the E effect.
Is the dose trend explained by prompt length?	P0 / P50 / E token counts	Mean prompt tokens decrease as TaskPass rises	The observed trend is not a simple prompt-length effect.
Does early termination create the gap?	Continue after rejection	E–P_REPEAT remains $+65.0$ pp	Stopping at an inapplicable action does not create the gap.
Does thinking alter the gap?	Thinking ON, matched tasks	P0, P_REPEAT, and E each pass 60/60	Enabling thinking removes the observed success-rate gap in this comparison.
Is the effect specific to BF16 or FP8?	BF16 vs. FP8	Both show large E–P0 gains	The content effect appears at both precisions.

C.3 SAME-STATE ACTION TAXONOMY

In TextWorld, same-state analysis follows paired executions while their public states remain identical and stops at the first differing action. In ALFWorld, it compares the next action from each of 100 shared initial states. The directional difference in percentage points is

$$100 \times \frac{\#\texttt{DIRECT_RESCUE} - \#\texttt{REVERSE}}{N}. \tag{7}$$

Table 9: Same-state action taxonomy used in the analysis.

Class	Condition
SAME	Same parsed operation, both applicable, equal next state.
DIRECT_RESCUE	Baseline action inapplicable, exposed action applicable.
REVERSE	Baseline action applicable, exposed action inapplicable.
BENIGN_BRANCH	Both actions applicable but select different operations.
BOTH_INAPPLICABLE	Both selected actions are inapplicable.

The repetition comparison used for same-state analysis is separate from the task-success comparison in Figure 3B. Across its 60 paired tasks, the taxonomy yields 39 direct rescues, zero reverse cases, four benign branches, one both-inapplicable case, and 16 pairs classified as SAME. The direct-minus-reverse score is $+65.0$ pp with a 95% task-paired bootstrap interval of $[53.33, 76.67]$ pp.

Derivation depth. Depth is the minimum Horn-rule proof height, with base facts at depth zero. All 60 preparation tasks have maximum reachable depth seven, 12 derived predicates, and 14 rules. These task-level quantities therefore provide no variation for stratification. Among the 40 first-divergence states, direct rescues occur in 4/4, 25/26, 2/7, and 0/3 cases at depths 1, 3, 4, and 5, respectively. The remaining 20 task pairs have no divergence and receive no divergence depth. These strata show no increasing rescue rate with depth. Because the divergence state depends on earlier actions, they do not identify a causal effect of depth.

C.4 CONCRETE SAME-STATE CASES

Direct rescue. Both executions have completed observation while three preparations remain pending. P_REPEAT repeats the completed inspection and is inapplicable, whereas E selects an applicable preparation. Both trajectories later fail, showing that an immediate action rescue need not imply eventual task success.

Legal branch. After observation and preparation are complete, both next actions are legal. P_REPEAT selects an irreversible failure operation, while E selects an unlock operation and later

Table 10: Representative same-state action differences in TextWorld.

Case	P_REPEAT action	Explicit action	Task outcome (P_REPEAT / E)
1	Repeats a completed inspection	Selects an applicable preparation operation	Fail / Fail
2	Selects an applicable operation that causes irreversible failure	Selects unlock	Fail / Pass

Both actions in Case 2 are applicable. Selecting an applicable action in Case 1 does not lead to task success.

succeeds. Same-state analysis therefore captures behavioral differences that are not limited to applicability errors.

Both inapplicable. Both executions repeat the completed inspection from the same state and fail under early termination.

C.5 Reasoning, executor, and environment comparisons

The effect varies across executors, inference settings, and environments. In TextWorld, P0, P_REPEAT, and E each complete all 60 tasks of Figure 3B with thinking enabled, while the same tasks show large exposure effects with thinking disabled. In a separate calibration with thinking enabled, Explicit also completed 10/10 tasks at the highest difficulty level considered. In the BabyAI comparison of Figure 3D, the E-D difference is 15 percentage points at the lower generation limit and zero at the higher limit. Both conditions improve, but the paired interval for the change in their difference includes zero. The cross-executor and three-condition budget comparisons are reported separately in Appendix C.7. DeepSeek and GLM show reverse-signed TaskPass estimates in their BabyAI studies. The 8K exact paired test and the paired interaction interval do not establish a change in the BabyAI E–D difference. The reverse-signed DeepSeek and GLM estimates do not establish harm.

The ALFWorld fixed-state comparisons use the same 100-state design. Qwen BF16 thinking OFF is positive ($+8$ pp) and Gemma 4 31B IT BF16 also shows a positive directional effect ($+10$ pp), while DeepSeek is close to zero ($+1$ pp). The Qwen thinking-ON observed-output estimate is $+4$ pp, but truncation prevents an unqualified comparison with OFF.

TextWorld token use with thinking enabled. In the 60-task P0/P_REPEAT/E comparison, all three conditions succeed on every task with seven action calls and no output truncation. Neutral padding matches the first-prompt length within each task. Recorded cumulative input use is also identical across conditions for every task, averaging 41,808.4 tokens. Mean completion tokens per task are 7,862.5 for P0, 8,122.7 for P_REPEAT, and 8,626.8 for E. The paired E–P0 and E–P_REPEAT differences are therefore $+764.3$ and $+504.1$ tokens per task. Exploratory task-paired percentile bootstrap intervals are $[378.3, 1156.0]$ and $[140.7, 872.4]$, respectively. These pointwise 95% intervals use 100,000 resamples with seed 1088. These results show no reduction in recorded completion use from providing the derived state in this comparison. With thinking enabled, the matched control therefore supports the interpretation that sufficient model-side computation can eliminate the success-rate difference, but it does not provide evidence of lower completion use from system-side derivation. The usage records do not separate reasoning tokens from other output. Usage statistics cover returned responses.

C.6 Preparation control and reconstruction outcomes

TaskPass is D=21/60, C=60/60, R=42/60, and E=58/60. The primary paired contrasts are reported in Table 11.

The E/D same-state comparison gives 31 direct rescues and zero reverse cases. R reconstructs 202/395 registries exactly (51.14%), with 4,168/4,740 individual entries correct (87.93%). No entry is UNKNOWN. All 18 unsuccessful R trajectories have an incorrect registry at the terminal decision and at the first decision after which the goal is no longer reachable within the remaining action budget. The first incorrect registry occurs at step three in 14 tasks and step five in four. In the one-step

Table 11: Preparation-control primary contrasts. Effects are percentage points.

Contrast	Effect	95% CI	Holm p
E–D	+61.67	$[50.00, 73.33]$	4.37×10^{-11}
R–C	-30.00	$[-41.67, -18.33]$	1.53×10^{-5}
E–R	+26.67	$[13.33, 40.00]$	4.02×10^{-4}

diagnostic, replacing the first mismatched registry changes action applicability in three tasks, with two improvements and one deterioration. The replacement response is not executed or propagated to later decisions, so this comparison measures immediate applicability and does not measure the outcome of a fully corrected trajectory.

Preparation notes. A deterministic lexical analysis finds predicate names with explicit TRUE/FALSE assignments in 35/420 C notes using complete identifiers and 298/420 (71.0%) when task-specific aliases with the random namespace removed are also included. The latter patterns occur in all 60 tasks and in 197/273 notes from the 39 tasks where D fails and C succeeds. No note contains an exact operation identifier, but preparation receives no operation catalog. These counts describe assignment patterns without determining their correctness or excluding natural-language planning.

Procedure without preparation. A subsequent execution adds an adapted description of the public closure procedure to each D action request on the same 60 tasks. It retains the closure steps and defines FALSE as non-membership in the closure, while omitting the registry-output instruction and ordered registry schema. The original system message, action tools, task data, decoding settings, and seven-action horizon are retained. No derived values or preparation call are added. This condition completes 20/60 tasks, compared with 21/60 in the original D execution. The paired difference is -1.7 percentage points with a 95% percentile bootstrap interval of $[-8.3, +5.0]$ from 100,000 task resamples with seed 2026092601. Both executions succeed on 18 tasks, only the new condition succeeds on two, and only the original D succeeds on three. The exact two-sided McNemar p is 1.0. All 281 responses return valid tool calls without truncation. This procedure description alone does not reproduce the preparation gain. The comparison uses independent executions on matched tasks and does not establish equivalence.

Table 12 reports actual token use in this comparison. The mean first request uses 4,720.1 prompt tokens under D and 5,021.9 under E. The added registry costs 301.8 tokens on average, with a mean within-task share of 6.02% of E's first prompt. Episode totals vary with trajectory length because early failures reduce the number of subsequent calls.

Table 12: Token use in the preparation comparison.

Condition	TaskPass	Prompt	Completion	Total	Calls
D	21/60	22,976	172	23,148	4.82
E	58/60	34,915	247	35,162	6.95
C	60/60	54,291	3,896	58,187	14.00
R	42/60	54,315	2,550	56,865	13.17

Tokens and calls are means over all 60 task attempts, including failures. Counts include preparation and action calls. R's totals exclude the additional corrected-registry calls used only for diagnosis.

C.7 BABYAI THINKING-ON TERMINATION

In the E/D budget comparison of Figure 3D, 35 of 39 task failures occur before the next action is produced. This comparison evaluates both conditions at 8K and 16K on the 60 tasks shared with the cross-executor comparison.

At 8K, E=50/60 and D=41/60 ($+15.0$ pp, paired $n_{11} = 34, n_{10} = 16, n_{01} = 7, n_{00} = 3$, exact two-sided McNemar $p = 0.093$, task-paired bootstrap 95% CI $[0, +30]$ pp). At 16K, E=D=55/60 (paired $n_{11} = 51, n_{10} = 4, n_{01} = 4, n_{00} = 1$, exact $p = 1.0$, 95% CI $[-10, +10]$ pp). The direct same-task interaction, $\Delta_{16K} - \Delta_{8K}$, is -15.0 pp with a 100,000-resample task-paired bootstrap 95% CI of $[-31.7, +1.7]$ pp. The interval includes zero, leaving the magnitude of the budget effect uncertain.

D at 8K has 19 truncations before the next action. E at 8K has nine truncations and one other termination without a tool call. D at 16K has four truncations and one other termination without a tool call. E at 16K has one truncation and four task failures after valid actions. These mutually exclusive categories comprise 33 truncations, two other terminations without a tool call, no parser-invalid tool calls, and four failures after valid actions.

Restricting to pairs in which both responses were valid and nontruncated gives descriptive E/D TaskPass of 34/34 versus 34/34 at 8K ($N = 34$), and 51/54 versus 54/54 at 16K ($N = 54$). These subsets are selected after execution and do not estimate a causal effect.

The Qwen cross-executor comparison uses the same 60 task identities in separate executions at 8K and reports E=52/60 and D=36/60 ($+26.7$ pp). The three-condition budget comparison includes P_REPEAT and uses 60 different tasks with no overlap. It reports E=43/60 and D=41/60 at 8K ($+3.3$ pp), and E=59/60 and D=55/60 at 16K ($+6.7$ pp). The change in the observed E–D difference therefore varies across the two budget comparisons.

C.8 Exploratory budget comparison

A separate exploratory comparison uses 16 BabyAI tasks. At a completion limit of 8,192 tokens, D, P_REPEAT, and E pass 10, eight, and 11 tasks, respectively. At 16,384 tokens, they pass 14, 14, and 15. This sample is analyzed separately from the 60-task comparison in Figure 3D.

C.9 ALFWorld thinking-ON truncation sensitivity

The Qwen BF16 thinking-ON fixed-state study contains 100 paired states. Its generation limit is 16,384 tokens, compared with 4,096 when thinking is disabled. The ON–OFF difference therefore compares configurations that vary both thinking and generation limit. Forty-eight truncated cells occur across 42 pairs and remain in the observed-output analysis. Under this scoring, there are four direct rescues and zero reverse cases, giving $S_{ON} = +4$ pp with bootstrap 95% CI $[1, 8]$ and exact two-sided sign $p = 0.125$. The paired ON-minus-OFF effect is -4 pp with 95% CI $[-10, 1]$.

Applying the action parser to the existing truncated responses leaves 32 pairs without a resolved latent action. The corresponding deterministic bounds are $S_{ON} \in [-4, 31]$ pp and $S_{ON} - S_{OFF} \in [-12, 23]$ pp.

C.10 ALFWorld task completion

The Qwen comparison uses the same 100 tasks and initial states as the same-state study in a separate execution, with thinking disabled, an 18-action horizon, and a generation limit of 4,096 tokens per action. Explicit succeeds on 16 tasks and Derivable on one. Only Explicit succeeds on 16 tasks, only Derivable on one, and neither succeeds on 83. The paired difference is 15.0 percentage points, with a 95% percentile bootstrap interval of [8.0,23.0] from 100,000 task resamples using seed 2026092601. The exact two-sided McNemar p value is 0.000275.

All 3,404 responses return parsed tool calls without truncation or technical failure. All 84 Explicit and 99 Derivable failures exhaust the action budget. In 15 and 22 of these trajectories, respectively, every action is applicable. Mean total token use per task is 727,158 under Explicit and 788,020 under Derivable, including unsuccessful episodes. These totals include input and output tokens and vary with trajectory length.

The Qwen and DeepSeek task comparisons use the same tasks, initial states, action interface, and budget. Their inference services and treatment of technical failures differ.