SKILL CLEANLINESS: DIAGNOSING AND RESTORING EXECUTION RECOVERABILITY IN AGENT SKILLS

Anonymous authors
Paper under double-blind review

ABSTRACT

Skills provide language-model agents with reusable knowledge and operating procedures. Yet an agent can fail even when a Skill contains correct knowledge sufficient to solve the task. We introduce *Skill Uncleanliness* to describe organizational barriers that make task-relevant knowledge difficult to extract or available knowledge difficult to turn into executable actions. To diagnose these barriers, we propose Skill Cleanliness Score (SCS), which evaluates how knowledge can be accessed and used for a particular task and executor, and localizes defects that obstruct execution. We further propose Verified Skill Restoration (VSR), which uses the diagnosis to expose relevant evidence and make execution relations explicit while retaining their source support, intervening only when the repair has verifiable grounding. Experiments across task environments and model families show that targeted restoration can improve task success. On ScienceWorld, VSR improves success by 4.8–9.3 percentage points over a raw demonstration overlay with the same execution gate across seven executors from four model families. On ALFWorld, observation-driven progress restoration substantially improves execution by smaller models.

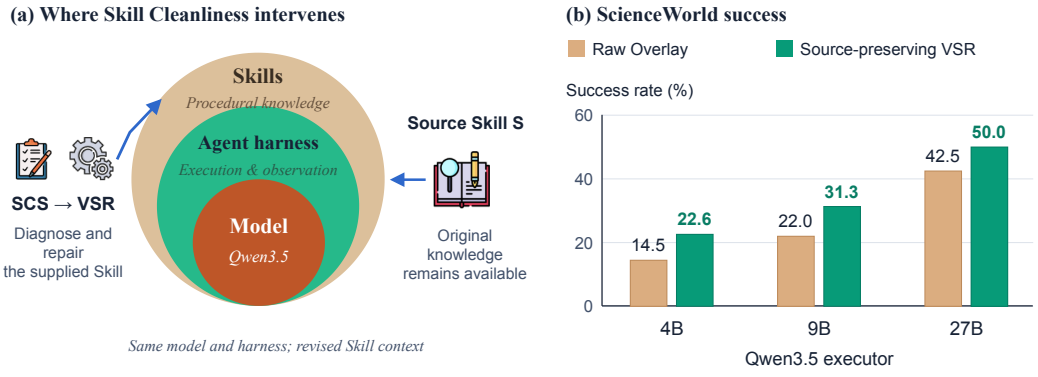


Figure 1: Skill-layer diagnosis and restoration. The Qwen3.5 subset illustrates ScienceWorld gains under a shared source library and gate; Table 2 extends this comparison to seven executors.

1 INTRODUCTION

Skills provide language-model agents with reusable knowledge and operating procedures. Recent systems acquire and refine Skills, while emerging benchmarks evaluate their generation and retrieval at scale (Wang et al., 2023; Shinn et al., 2023; Su et al., 2026; Kang et al., 2026; Zhou et al., 2026; Chen et al., 2026). Their utility depends on how effectively an executor can turn the supplied knowledge into action.

A Skill can contain correct and sufficient knowledge yet remain difficult to execute. Consider a demonstration that correctly moves an animal into a red box in the kitchen. A new task asks for an animal in a yellow box in the bathroom. The procedure is available, but following it requires

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.

separating reusable actions from the example's destinations and binding those actions to the current task. Similarly, a complete procedure can leave an agent repeatedly searching the same locations when it cannot recover what has already been done. Both tasks require the executor to recover relations that the Skill leaves implicit.

We call such task- and executor-dependent organizational barriers *Skill Uncleanliness*. They arise when *task-relevant knowledge is difficult to extract* or *available knowledge is difficult to turn into executable actions*. Cleanliness therefore concerns usable organization: the same evidence may be essential for one task and distracting for another, and a relation that one executor readily infers may remain inaccessible to another. We use *execution recoverability* to describe the ability to recover the relations needed to act from a supplied Skill.

This perspective complements work on acquiring procedures and memory (Li et al., 2025; Jiang et al., 2026a; He et al., 2026), selecting external knowledge and tools (Lewis et al., 2020; Schick et al., 2023; Qin et al., 2024; Patil et al., 2024), and organizing reasoning and planning (Yao et al., 2023b; Wei et al., 2022; Zhou et al., 2023; Yao et al., 2023a). Our question is how to identify and reduce execution barriers within an already available Skill while retaining the knowledge supporting the task.

We address this question through diagnosis followed by restoration. Skill Cleanliness Score (SCS) localizes barriers to recovering evidence and execution relations. Verified Skill Restoration (VSR) makes the corresponding knowledge accessible. Diagnosis determines what to repair; verification and an executor-specific deployment decision determine whether to intervene. Our contributions are:

- **Skill Uncleanliness.** We formulate organizational barriers to using correct, task-sufficient knowledge and characterize them through evidence access and recovery of executable relations.
- **SCS diagnosis.** We operationalize cleanliness as a four-dimensional profile with source-linked defect certificates that identify where targeted restoration is warranted.
- **VSR restoration.** We develop diagnosis-guided repairs that make execution relations explicit, check their source support and structure, and retain the original Skill when acceptance conditions are unmet.

Experiments test both the diagnosed barriers and their matched repairs. ScienceWorld results across four model families and seven executors support task-binding restoration; ALFWorld tests online progress recovery, and MedCalc tests evidence selection.

2 Related Work

Reusable agent procedures connect skill acquisition, external memory, and structured action. Skill libraries and memory systems retain experience for later tasks (Wang et al., 2023; Shinn et al., 2023; Park et al., 2023; Wu et al., 2023; Packer et al., 2023). Recent work studies skill retrieval, generation, composition, and lifecycle organization (Su et al., 2026; Kang et al., 2026; Zhou et al., 2026; Chen et al., 2026; Jiang et al., 2026b). These directions motivate diagnosing how a supplied procedure is consumed.

Access and execution remain distinct evaluation targets. Retrieval and tool-use methods improve selection or invocation (Lewis et al., 2020; Schick et al., 2023; Qin et al., 2024; Patil et al., 2024), while long-context results expose failures to use available evidence (Liu et al., 2024a). Interactive benchmarks evaluate grounded actions and consequential state changes (Shridhar et al., 2021; Wang et al., 2022; Trivedi et al., 2024; Zhou et al., 2024; Liu et al., 2024b); newer evaluations broaden tool and software-agent settings (Yan et al., 2025; Chen et al., 2025; Gao et al., 2025; Zhang et al., 2025). Our experiments diagnose representation-level burdens within fixed execution settings.

The framework also connects to partial observability, temporal abstraction, information selection, and verified transformation (Kaelbling et al., 1998; Sutton et al., 1999; Tishby et al., 1999; Leroy, 2009; Solar-Lezama, 2008; Weimer et al., 2009). Skill Cleanliness links a source-auditable diagnosis to a specific restoration and fallback decision.

3 METHOD

Our approach diagnoses organizational barriers, exposes the evidence or relations needed for execution, and verifies the intervention before deployment. Figure 2 summarizes how SCS diagnosis guides VSR restoration.

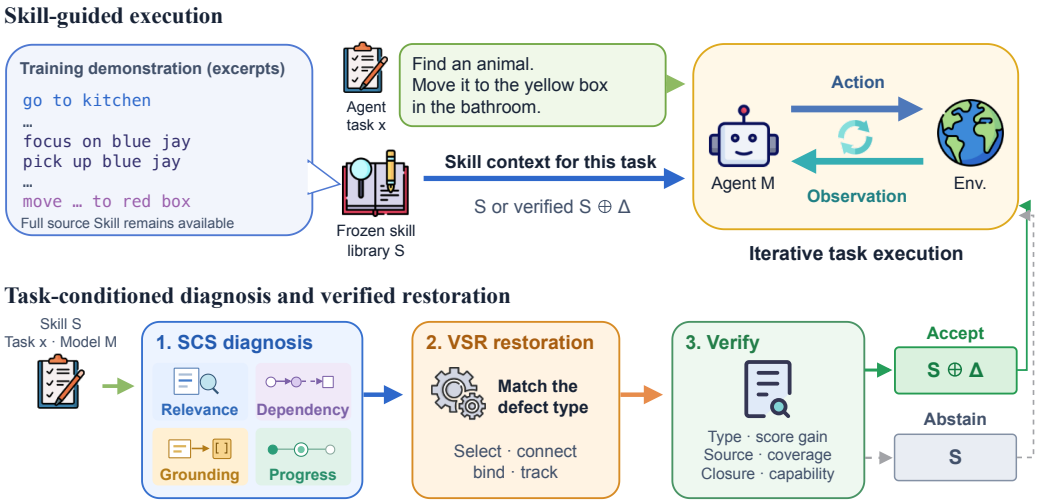


Figure 2: From diagnosis to restoration. SCS locates barriers in a supplied Skill; VSR makes the corresponding execution relations explicit. The verified overlay retains the source library, while observations resolve runtime bindings and update progress.

3.1 SKILL UNCLEANLINESS AND EXECUTION RECOVERABILITY

Let S be a Skill, x a task, and M an executor, including its model, context budget, decoding policy, and action and observation interfaces. Semantic correctness means that the Skill's facts, rules, and action effects are valid under an ideal interpreter. Task sufficiency means that its knowledge supports a solution:

$$\text{Sufficient}(S, x) = 1 \iff K(S) \models \exists \pi \, [\pi \text{ satisfies } x]. \tag{1}$$

Neither property specifies how easily M can recover the evidence and relations needed to execute that solution. We define *Skill Uncleanliness* as organizational barriers to this recovery for (S, x, M). The two broad burdens are accessing relevant knowledge and converting it into executable actions; they motivate the diagnostic dimensions below.

For (S, x), let $G^*(S, x) = (V, E, B, O, T)$ denote an ideal execution graph: V contains obligations, actions, and intermediate states; E contains producer–consumer dependencies; B contains bindings; and O, T contain observable postconditions and termination predicates. SCS approximates this graph from source spans in the Skill and task; runtime observations supply unresolved values.

3.2 SCS: DIAGNOSING BARRIERS FROM KNOWLEDGE TO ACTION

SCS returns

$$\text{SCS}(S, x, M) = (C_{rel}, C_{dep}, C_{grd}, C_{prog}). \tag{2}$$

Relevance evaluates whether required evidence is covered and distinguishable from irrelevant material. The remaining dimensions examine its conversion into action: *Dependency* measures how explicitly intermediate products connect to subsequent steps; *Grounding* assesses the availability and ambiguity of explicit, source-supported parameter values; and *Progress* measures whether observations establish what is complete and what remains to be done.

A semantic parser identifies supporting source spans and graph objects; deterministic procedures compute the scores. Each diagnosed defect carries its type, affected objects, and evidence in a

Algorithm 1 SCS diagnosis and VSR for a target defect type

Require: Skill S, task x, executor M, target type k, frozen $(\tau_k, \gamma_{M,k})$
Ensure: Original S or a verified task-local intervention $S \oplus \Delta$
$G \leftarrow \mathrm{Extract}(S, x)$; $\mathbf{C} \leftarrow \mathrm{Score}(G)$
$c \leftarrow \mathrm{Certify}(G, k)$ ▷ issue, affected objects, source spans
if $c = \perp$ **or** $C_k = \mathrm{N/A}$ **or** $100C_k \geq \tau_k$ **or** $\gamma_{M,k} = 0$ **then**
  return S
end if
$p \leftarrow \mathrm{PASS_TABLE}[k]$; $\Delta \leftarrow \mathrm{Propose}_p(S, x, c)$
if $\Delta \neq \varnothing$ **and** $\mathcal{V}_k(S, \Delta, c, M)$ **then** ▷ Eq. 3
  return $S \oplus \Delta$
end if
return S

certificate. In the animal example, the certificate links the demonstration's destinations to the values required by the current task. Inapplicable dimensions are marked N/A. The four-dimensional profile guides repair; an optional scalar summarizes it. Detailed score definitions appear in the supplement.

3.3 VSR: RESTORING RECOVERABLE EXECUTION

VSR uses the certificate to expose knowledge the executor would otherwise have to reconstruct. Evidence selection brings necessary source material into focus; dependency and binding repairs connect actions to their supported inputs. For interactive tasks, an observation-driven frontier records confirmed progress and exposes ready steps. In the running example, task-supported room and container substitutions make the demonstration usable, while the animal's identity remains unresolved until observed.

We write $S \oplus \Delta$ for the execution representation after a task-local intervention: a source-supported selection, an overlay, or a runtime interface. Selected evidence is retained verbatim; source-preserving overlays keep the full library available; progress repairs update the interface online. These environment-specific backends share a certificate and acceptance interface.

Algorithm 1 gives the per-defect procedure. Development data freeze a capability flag $\gamma_{M,k}$ and threshold τ_k for repair type k; inapplicable dimensions do not trigger. Relevance certification additionally requires evidence precision below 0.8. Candidates undergo score, coverage, and structural checks:

$$\begin{aligned}\mathcal{V}_k = \mathrm{Match}(c, \Delta, k) &\wedge [C_k(S \oplus \Delta) > C_k(S)] \\ &\wedge [\mathrm{Coverage}(S \oplus \Delta) \geq \mathrm{Coverage}(S)] \\ &\wedge \mathrm{Provenance}(c, \Delta) \wedge \mathrm{Closed}(\Delta) \wedge \mathrm{Fits}_M(S \oplus \Delta).\end{aligned} \tag{3}$$

Coverage compares the aggregate support for the same extracted obligations. Closure checks prerequisite providers, including for pending runtime values; Fits_M checks interface and budget compatibility. The output is

$$\mathrm{Output}(S, c, M) = \begin{cases} S \oplus \Delta, & \mathrm{Verify}(\Delta, c, M) = 1, \\ S, & \mathrm{Verify}(\Delta, c, M) = 0. \end{cases} \tag{4}$$

4 THEORY: RECOVERY AND REPAIR CONDITIONS

This explanatory model separates available knowledge from executable recovery and states sufficient conditions for beneficial repair.

Condition on (S, x, M). Let E_j denote correct recovery of relation r_j before use and $E = \cap_j E_j$, with a fixed order. Assume every admitted successful plan requires E. When $P(E) > 0$, define $q_j = P(E_j \mid E_{<j})$ and residual competence $\kappa = P(\mathrm{success} \mid E)$. The chain rule gives

$$X_M(S, x) = \kappa \prod_{j=1}^{m} q_j = \kappa e^{-B}, \qquad B = \sum_{j=1}^{m} -\log q_j. \tag{5}$$

Table 1: Qwen3.5 selective recovery over all evaluated inputs. N counts task–representation inputs for the two balanced mixtures, and test variations for ScienceWorld. Gains are percentage points. Mixture results are computed from the fixed-condition outcomes.

Study	Model	N	No repair (%)	Routed VSR (%)	Gain
MedCalc mixture	Qwen3.5-4B	1,600	55.3	61.9	+6.6
	Qwen3.5-9B	1,600	67.4	69.6	+2.2
	Qwen3.5-27B	1,600	79.1	79.1	+0.0
ALFWorld mixture	Qwen3.5-4B	268	53.4	84.3	+31.0
	Qwen3.5-9B	268	59.3	82.1	+22.8
	Qwen3.5-27B	268	88.8	96.3	+7.5
ScienceWorld test	Qwen3.5-4B	332	16.3	22.6	+6.3
	Qwen3.5-9B	332	23.2	31.3	+8.1
	Qwen3.5-27B	332	37.7	50.0	+12.3

If $P(E) = 0$, success is zero directly; no conditional κ is needed. Here B is the negative log-probability of recovering all required relations.

Proposition 1 (A bounded-inspection construction)**.** *In a finite-inspection model, task-sufficient Skills with identical task-relevant knowledge can have arbitrarily small execution success despite all included statements being true.*

Proof. Fix $1 \le k \le b$ and place k required source units among $N \ge b$ true units. An executor uniformly samples b units without replacement and requires all k. With conditional execution probability $\kappa_N \in [0, 1]$,

$$X_M(S_N, x) = \kappa_N \frac{\binom{N-k}{b-k}}{\binom{N}{b}} = \kappa_N \frac{(b)_k}{(N)_k} \longrightarrow 0 \quad \text{as } N \longrightarrow \infty. \tag{6}$$

True irrelevant additions preserve task sufficiency while reducing recovery. For $b < k$, success is zero. □

Proposition 2 (Conditional restoration benefit)**.** *For the same ordered necessary relations, suppose* $P(E) > 0$ *under both representations. If* $q'_j \ge q_j$ *for every* j *and* $\kappa' \ge \kappa$*, then* $X_M(S \oplus \Delta, x) \ge X_M(S, x)$*. When the baseline success probability is positive,*

$$\frac{X_M(S \oplus \Delta, x)}{X_M(S, x)} = \frac{\kappa'}{\kappa} \exp(B - B'). \tag{7}$$

Multiplying the nonnegative factors in Eq. 5 proves the result. The conditions concern latent recovery probabilities; repair utility is evaluated on held-out tasks.

5 Experiments

Our experiments connect each design choice to execution: SCS localizes barriers, matched VSR repairs expose usable evidence and relations, and verification supports selective deployment. We evaluate recovery across executors, explain the repairs through controlled ablations, and trace their operation in execution cases.

5.1 Selective restoration improves execution

The evaluated executors are listed in Table 2; the core representation studies use its Qwen executors. Development choices are frozen before held-out evaluation. MedCalc evaluates held-out questions, ALFWorld uses the `valid_unseen` split, ScienceWorld tests task variations with a training-derived demonstration library, and AppWorld evaluates API workflows. Reported success rates retain failed executions and fallback outcomes. The supplement specifies decoding, development splits, and environment protocols.

Table 2: ScienceWorld success (%) on the same 332 variations. Raw Overlay and VSR share the frozen library and executor-gate rule; VSR adds verified task binding. Gains and paired p values compare VSR with Raw.

Executor	No repair	Raw Overlay	VSR	Gain (pp)	Paired p
Qwen3.5-4B	16.3	14.5	22.6	+8.1	3.55×10^{-4}
Qwen3.5-9B	23.2	22.0	31.3	+9.3	1.52×10^{-4}
Qwen3.5-27B	37.7	42.5	50.0	+7.5	1.87×10^{-3}
Gemma-4-12B	31.9	36.4	44.3	+7.8	1.90×10^{-3}
Gemma-4-31B	30.1	33.7	42.8	+9.0	5.30×10^{-6}
Gemini 3.8 Flash	51.8	51.5	56.3	+4.8	4.02×10^{-2}
DeepSeek V4.1 Flash	31.6	32.8	40.7	+7.8	2.94×10^{-3}

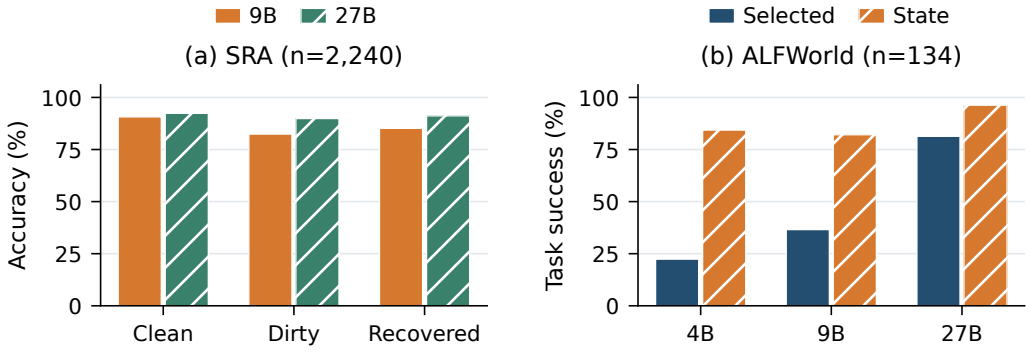


Figure 3: Restoring executable relations. Left: controlled SRA dependency recovery, with 2,240 task-seed observations per condition/model. Right: ALFWorld progress recovery from Selected to State, with 134 tasks per condition/model.

Selective restoration improves execution while preserving already usable inputs (Table 1). MedCalc pairs each question with Full and Selected evidence; ALFWorld pairs each task with static Selected and an already restored State interface. The mixtures contain 1,600 and 268 task–representation inputs. For MedCalc, the 4B/9B routers improve accuracy by 6.6/2.2 pp; the frozen 27B gate retains the original representation. For ALFWorld, routing improves success by 31.0, 22.8, and 7.5 pp. Table 4 separately records detection, verification, and preservation, including preservation of every Selected MedCalc input.

We use exact McNemar tests for paired outcomes, Skill-level sign-flip tests for SRA, and clustered bootstrap intervals for Skill- or family-level uncertainty. The supplement reports the full protocols, statistical units, and uncertainty estimates.

5.2 Verified task binding transfers across executors

Binding makes retrieved procedures executable. VSR exceeds Raw Overlay by 4.8–9.3 pp across all seven executors from four model families (Table 2). Raw and VSR share the frozen demonstration library and executor-gate rule, making this comparison directly relevant to the Grounding repair: VSR connects the retrieved procedure to task-supported instance values. For Qwen3.5-4B/9B, Raw Overlay gives 14.5%/22.0% success, whereas binding raises success to 22.6%/31.3%. Gains on Gemma, Gemini, and DeepSeek support transfer across executor capabilities. All seven episode-level paired tests reach nominal significance. The supplement reports task-family uncertainty and executor-specific development protocols.

5.3 MATCHED REPAIRS ADDRESS DISTINCT EXECUTION BARRIERS

Dependency: expose producer–consumer relations. Controlled SRA holds task knowledge and answers fixed while making dependencies harder to recover through aliases, cross-section references, and implicit state. Figure 3 shows lower accuracy under Dirty than Clean for both executors, followed by partial recovery when the relations are made explicit again. Recovered remains below Clean. This controlled construction supports targeting relations that connect available facts to subsequent steps.

Relevance: focus evidence while retaining its support. MedCalc provides a direct test of evidence selection (Table 3). Moving from Full to Selected raises 4B accuracy from 46.6% to 64.0% and 9B accuracy from 64.5% to 70.2%. Selected evidence also exceeds a static contract at every scale. Removing source evidence from the contract lowers accuracy to 52.1%, 56.9%, and 66.1%, supporting VSR's requirement to preserve the evidence behind an execution representation. Here the effective repair is to expose the relevant source units, consistent with the Relevance diagnosis.

Progress: expose what observations have established. ALFWorld requires the agent to track completed actions and determine which steps are ready. State adds this observation-driven interface and raises success from 22.4%/36.6%/81.3% under Selected to 84.3%/82.1%/96.3% for 4B/9B/27B (Table 3). Static contracts achieve 41.0%, 38.8%, and 82.1%; initial-token-matched source achieves 28.4%, 34.3%, and 81.3%. State exceeds both controls at every evaluated scale, supporting an online interface that exposes confirmed state and ready actions.

The paired outcomes make this progress-recovery effect more concrete (Table 5). For 4B, State solves 84 tasks that Selected misses and loses only one; the corresponding 9B comparison has 66 wins and five losses. Even for 27B, whose Selected success is already high, State adds 20 successes without losing a previously successful task. The benefit therefore reflects a consistent direction of change on the same tasks. Together with the static and token-matched controls in Table 3, this supports exposing observed completion and ready actions as part of the execution interface.

Typed diagnosis connects each barrier to its repair. Table 4 shows that SCS responds to the distinctions used by the restoration pipeline. It detects all controlled Dirty dependency inputs and all static progress inputs, without triggering on their Clean and State counterparts. Relevance certification preserves every Selected Skill, and Grounding scores increase on all 291 applicable candidates, with 78.0% accepted after verification. This supports the typed profile as a repair interface for these constructions.

5.4 VERIFICATION SUPPORTS SELECTIVE DEPLOYMENT

The type-routing audit tests whether the diagnosis constrains the proposed operation (Table 4). Each certificate is paired with its matching compiler pass and with incompatible passes. All 269 matched pairs pass the shared kernel, while all 807 wrong-type pairs are rejected. This supports a concrete role for the defect type: it selects the admissible repair operation and excludes incompatible interventions.

Table 3: Qwen3.5 fixed-representation ablations. Entries are success rates (%). Contract-only removes source evidence; State adds observation-driven progress.

Benchmark	Model	Full	Selected	Contract	Contract-only	Frozen gate
MedCalc	Qwen3.5-4B	46.6	64.0	60.8	52.1	64.0
	Qwen3.5-9B	64.5	70.2	69.8	56.9	69.8
	Qwen3.5-27B	78.9	79.2	76.5	66.1	78.9
		Selected	Contract	Contract-only	State	Token-matched
ALFWorld	Qwen3.5-4B	22.4	41.0	38.1	84.3	28.4
	Qwen3.5-9B	36.6	38.8	42.5	82.1	34.3
	Qwen3.5-27B	81.3	82.1	82.1	96.3	81.3

Table 4: Diagnostic response rates (%) and type-routing checks. The final row evaluates the shared verification kernel, not task success.

Dimension	Paired construction	Diagnostic evidence
Relevance	MedCalc Full/Selected ($n = 40$ Skills)	87.5% Full flagged; 72.5% repairs verified; 100.0% Selected preserved
Dependency	SRA Dirty/Clean ($n = 7$ Skills)	100.0% Dirty detected; 0.0% Clean falsely triggered
Grounding	ScienceWorld Raw/task-bound ($n = 291$ candidates)	100.0% slot-score increases; 78.0% accepted certificates
Progress	ALFWorld static/State ($n = 6$ families)	100.0% static detected; 0.0% State falsely triggered
Type routing	Matched/wrong-type certificate–pass pairs	269/269 matched pairs pass; 807/807 wrong-type pairs rejected

Table 5: Paired execution controls with Qwen3.5. ALFWorld compares State with Selected on the full split; AppWorld compares Verified Contract with initial-token-matched source on the same accepted subset. Counts are successful tasks. W/L gives successes unique to repair/control; p is exact McNemar.

Study	Model	Control	Repair	W/L	Paired p
ALFWorld	4B	30/134	113/134	84/1	4.45×10^{-24}
	9B	49/134	110/134	66/5	1.19×10^{-14}
	27B	109/134	129/134	20/0	1.91×10^{-6}
AppWorld	9B	37/343	69/343	46/14	4.22×10^{-5}

Figure 4 (left) shows unchanged routed gains over thresholds 60–90 and positive gains at 95. The observed execution benefit therefore persists across these threshold settings, with Raw fallback retained in the evaluation.

Iterative repair expands the set of deployable representations in AppWorld (Figure 4, right). With audit repair enabled, the evaluated configurations attain higher coverage and higher full-set success than those with repair disabled. The structure-only configuration attains the highest observed coverage and slightly higher full-set success. These comparisons show that repair can expand coverage under the existing verifier, while relaxing verification yields a different coverage–success operating point.

5.5 Restoration extends to API workflows

AppWorld tests restoration in workflows involving authentication, runtime IDs, and repeated API side effects (Table 6). The frozen VSR gate improves full-set success over Official ReAct by 11.5, 17.7, and 29.5 pp for 4B/9B/27B, including source fallback for rejected candidates. It also exceeds BM25 Skill retrieval at all three scales. These full-set comparisons support applying the restoration system to API workflows.

Table 5 further examines the execution representation within a shared accepted subset. For 9B, Verified Contract solves 69/343 tasks, compared with 37/343 for source evidence repeated to match the initial Contract prompt length. The paired comparison gives 46 wins and 14 losses ($p = 4.22 \times 10^{-5}$). This control supports making execution relations explicit beyond supplying a comparable amount of initial context. Its shared subset separates the representation comparison from differences in which tasks pass verification; Table 6 reports the complete system with fallback.

The supplement reports operating costs, additional transfer studies, and applicability checks, including natural-Skill and visual studies without reliable gains and disagreement between AppWorld contract auditors.

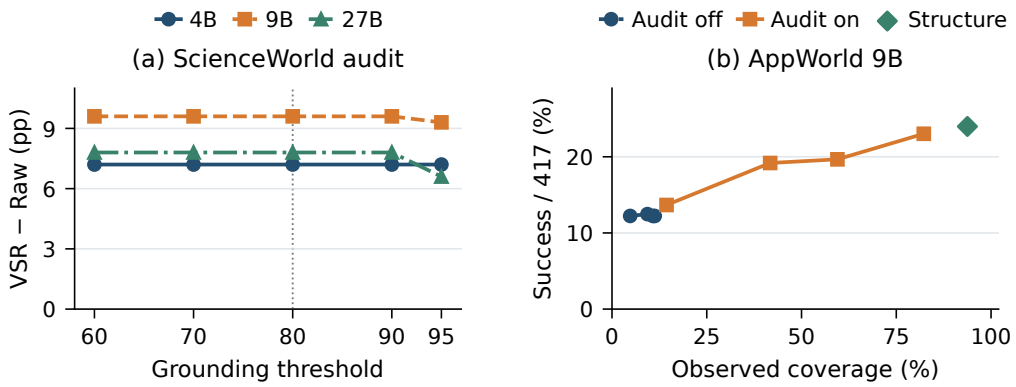


Figure 4: Robustness comparisons. Left: ScienceWorld success-rate gains of threshold-routed VSR over Raw Overlay on all 332 held-out test variations, using VSR outcomes for accepted cases and Raw fallback otherwise. The audit reuses fixed paired outcomes without new model execution. Right: AppWorld full-set success at observed verifier coverage, retaining all 417 tasks. Lines connect evaluated configurations.

Table 6: AppWorld success rates (%) over 417 tasks per Qwen3.5 executor. The VSR gate includes fallback.

Model	Official ReAct	BM25 Skill	Frozen VSR gate	Gain vs ReAct (pp)
Qwen3.5-4B	4.8	4.1	16.3	+11.5
Qwen3.5-9B	5.3	2.9	23.0	+17.7
Qwen3.5-27B	9.1	22.1	38.6	+29.5

Table 7: ScienceWorld `find-animal__v250` case with Qwen3.5-9B. Scores are points; the slot score is the Grounding certificate.

Observed quantity	Raw Overlay	Verified VSR
Room binding	Training instance: kitchen	Task text: bathroom
Container binding	Training instance: red box	Task text: yellow box
Slot certificate score	50.0	100.0
Episode score	−100.0 (failure)	100.0 (success)

5.6 Case analysis from diagnosis to execution

From a Grounding certificate to the correct action. The demonstration's red box and kitchen conflict with the task's yellow box and bathroom (Table 7). The Grounding certificate localizes both mismatches; VSR inserts task-supported substitutions while retaining the library, raising the slot score from 50.0 to 100.0. Animal identity remains pending until observations identify a common toad. VSR then completes `move common toad to yellow box`, while Raw repeatedly searches and fails. Task bindings and runtime observations together make the procedure executable.

6 Conclusion

In this paper, we introduce Skill Cleanliness to characterize barriers that prevent agents from using available knowledge, and propose verified skill restoration to address these barriers while preserving the source. Our experiments yield two findings. First, useful knowledge can remain difficult to execute because of irrelevant evidence, unresolved dependencies, missing progress information, or incorrect task-instance bindings. Second, repairs matched to these barriers can improve execution, but their benefits vary across tasks and executors.

AI USE STATEMENT

Generative AI tools assisted manuscript drafting and translation, as well as the formulation and exposition of the theoretical analysis (including proof revisions). The authors take responsibility for the final manuscript.

REPRODUCIBILITY STATEMENT

The supplement describes the evaluation protocols, statistical units, case examples, and diagnostic algorithms. The source package includes plotting data, scripts, and result ledgers. These support regeneration of the figures but do not constitute a complete execution-replication package: the underlying trajectories, environment versions, checkpoint identifiers, and seeds are not all included.

REFERENCES

C. Chen, X. Hao, W. Liu, X. Huang, X. Zeng, S. Yu, D. Li, S. Wang, W. Gan, and Y. Huang. Acebench: Who wins the match point in tool usage? *arXiv preprint arXiv:2501.12851*, 2025.

Shiqi Chen, Jingze Gai, Ruochen Zhou, Jinghan Zhang, Tongyao Zhu, Junlong Li, Kangrui Wang, Zihan Wang, Zhengyu Chen, Klara Kaleb, Ning Miao, Siyang Gao, Cong Lu, Manling Li, Junxian He, and Yee Whye Teh. Skillcraft: Can llm agents learn to use tools skillfully? *arXiv preprint arXiv:2603.00718*, 2026.

X. Gao, S. Xie, J. Zhai, S. Ma, and C. Shen. Mcp-radar: A multi-dimensional benchmark for evaluating tool use capabilities in large language models. *arXiv preprint arXiv:2505.16700*, 2025.

Zexue He, Yu Wang, Churan Zhi, Yuanzhe Hu, Tzu-Ping Chen, Lang Yin, Ze Chen, Tong Arthur Wu, Siru Ouyang, Zihan Wang, Jiaxin Pei, Julian McAuley, Yejin Choi, and Sandy Pentland. Memoryarena: Benchmarking agent memory in interdependent multi-session agentic tasks. *arXiv preprint arXiv:2602.16313*, 2026. URL https://arxiv.org/abs/2602.16313.

G. Jiang, Z. Su, X. Qu, and Y. R. Fung. Xskill: Continual learning from experience and skills in multimodal agents. *arXiv preprint arXiv:2603.12056*, 2026a.

Y. Jiang, D. Li, H. Deng, B. Ma, X. Wang, Q. Wang, and G. Yu. Sok: Agentic skills–beyond tool use in llm agents. *arXiv preprint arXiv:2602.20867*, 2026b.

Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1–2):99–134, 1998. doi: 10.1016/S0004-3702(98)00023-X.

Ryangkyung Kang, Hongcheol Cho, and Youngeun Kim. Skillret: A large-scale benchmark for skill retrieval in llm agents. *arXiv preprint arXiv:2605.05726*, 2026.

Xavier Leroy. A formally verified compiler back-end. *Journal of Automated Reasoning*, 43:363–446, 2009. doi: 10.1007/s10817-009-9155-4.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Kuttler, Mike Lewis, Wen-tau Yih, Sebastian Rocktaschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Fangzhou Li, Pagkratios Tagkopoulos, and Ilias Tagkopoulos. Skillflow: Scalable and efficient agent skill retrieval system. *arXiv preprint arXiv:2504.06188*, 2025. URL https://arxiv.org/abs/2504.06188.

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, 2024a. doi: 10.1162/tacl_a_00638.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, Minlie Huang, Yuxiao Dong, and Jie Tang. Agentbench: Evaluating llms as agents. In *International Conference on Learning Representations*, 2024b.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. Memgpt: Towards llms as operating systems. *arXiv preprint arXiv:2310.08560*, 2023.

Joon Sung Park, Joseph O'Brien, Carrie Cai, Meredith Ringel Morris, Percy Liang, and Michael Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, 2023. doi: 10.1145/3586183.3606763.

Shishir G. Patil, Tianjun Zhang, Xin Wang, and Joseph E. Gonzalez. Gorilla: Large language model connected with massive apis. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. Toolllm: Facilitating large language models to master 16000+ real-world apis. In *International Conference on Learning Representations*, 2024.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Noah Shinn, Federico Cassano, Ashwin Berman, Ashay Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Cote, 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://arxiv.org/abs/2010.03768.

Armando Solar-Lezama. Program synthesis by sketching. Technical Report UCB/EECS-2008-176, University of California, Berkeley, 2008.

Weihang Su, Jianming Long, Qingyao Ai, Yichen Tang, Changyue Wang, Yiteng Tu, and Yiqun Liu. Skill retrieval augmentation for agentic ai. *arXiv preprint arXiv:2604.24594*, 2026.

Richard S. Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1–2):181–211, 1999. doi: 10.1016/S0004-3702(99)00052-1.

Naftali Tishby, Fernando C. Pereira, and William Bialek. The information bottleneck method. In *Proceedings of the 37th Annual Allerton Conference*, 1999.

Harsh Trivedi, Tushar Khot, Mareike Hartmann, Ruskin Manku, Vinty Dong, Edward Li, Shashank Gupta, Ashish Sabharwal, and Niranjan Balasubramanian. Appworld: A controllable world of apps and people for benchmarking interactive coding agents. In *Proceedings of ACL*, pp. 16022–16076, 2024.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Arun Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023.

Ruoyao Wang, Peter Jansen, Marc-Alexandre Cote, and Prithviraj Ammanabrolu. Scienceworld: Is your agent smarter than a 5th grader? In *Proceedings of EMNLP*, pp. 11279–11298, 2022. doi: 10.18653/v1/2022.emnlp-main.775.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Westley Weimer, ThanhVu Nguyen, Claire Le Goues, and Stephanie Forrest. Automatically finding patches using genetic programming. In *Proceedings of the 31st International Conference on Software Engineering*, pp. 364–374, 2009. doi: 10.1109/ICSE.2009.5070536.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W. White, Doug Burger, and Chi Wang. Autogen: Enabling next-gen llm applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155*, 2023.

Y. Yan, S. Wang, J. Du, J. Yang, Y. Shan, Q. Qiu, X. Jia, X. Wang, X. Yuan, and X. Han. Mcpworld: A unified benchmarking testbed for api, gui, and hybrid computer use agents. *arXiv preprint arXiv:2506.07672*, 2025.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems*, volume 36, 2023a.

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*, 2023b. URL https://arxiv.org/abs/2210.03629.

L. Zhang, S. He, C. Zhang, Y. Kang, B. Li, C. Xie, J. Wang, M. Wang, Y. Huang, S. Fu, E. Nallipogu, Q. Lin, Y. Dang, S. Rajmohan, and D. Zhang. Swe-bench goes live! *arXiv preprint arXiv:2505.23419*, 2025.

Denny Zhou, Nathanael Scharli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Denny Choi, Ilya Sutskever, Quoc V. Le, and Ed H. Chi. Least-to-most prompting enables complex reasoning in large language models. In *International Conference on Learning Representations*, 2023.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. Webarena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations*, 2024.

Yifan Zhou, Zhentao Zhang, Ziming Cheng, Shuo Zhang, Qizhen Lan, Zhangquan Chen, Zhi Yang, Qianyu Xu, Ronghao Chen, Huacan Wang, and Sen Hu. Skillgenbench: Benchmarking skill generation pipelines for llm agents. *arXiv preprint arXiv:2605.18693*, 2026.

A EXPERIMENTAL INVENTORY AND REPORTING PROTOCOL

This appendix presents fixed-representation experiments, selective-routing analyses, diagnostic checks, robustness studies, execution costs, negative results, and execution examples. Routing analyses reuse paired outcomes from the corresponding fixed-representation evaluations. Accepted-contract and full-task accuracy are reported with their respective denominators.

Table 8: Experimental inventory. Representation mixtures reuse paired inputs from the corresponding fixed-representation evaluations.

Study	Evaluation construction	Reporting unit
MedCalc fixed	300 development and 800 held-out questions; three model scales	Question; Skill cluster
MedCalc routing	40 held-out Skills × 20 questions × Full/Selected; 1,600 inputs/model	Paired representation input
ALFWorld fixed	All 134 `valid_unseen` tasks; four representations	Environment task
ALFWorld routing	The same 134 tasks paired with static Selected and State; 268 inputs/model	Reused paired input
Controlled SRA	7 Skills × 40 questions × 8 seeds; 2,240 per condition	Skill cluster
SRA routing	Clean/Dirty mixture of those outcomes; 4,480 inputs/model	Reused paired input
ScienceWorld	332 test variations from 19 families; training library covers 30 families	Variation; family cluster
AppWorld	143 development tasks; 343 accepted plus 74 fallback test tasks	All 417 test tasks
Natural SRA	19 Skills × 39 held-out questions × 8 seeds = 5,928	Skill cluster
CHAMP/TheoremQA	98 provided Skills; 2,144 paired seed samples	Skill cluster
Visual ALFWorld	134 tasks × 4 conditions = 536 episodes	Environment task
Surface dose	12 perturbation conditions × 134 tasks = 1,608 episodes	Environment task

The main execution studies use local Qwen3.5-4B, 9B, and 27B checkpoints, temperature zero, and thinking disabled. The visual study instead uses Qwen3-VL-8B-Instruct and is documented separately in Section I. Within each comparison, the task list, action protocol, step budget, and native evaluator are held fixed. Development outcomes select an executor-level representation or enable a repair type; held-out outcomes are not inputs to the router. Verification failures remain in the evaluation denominator and use the specified source fallback.

Success differences are paired on the same questions or environment tasks. McNemar tests are two-sided exact tests on discordant outcomes; wins/losses list successes unique to the first/second condition. The final evaluation protocol uses 20,000 Skill- or task-family bootstrap resamples with percentile endpoints at 2.5% and 97.5%. Earlier MedCalc/AppWorld gate-stability analyses retain their respective intervals and cluster definitions. The small-Skill SRA studies additionally use exact Skill-level sign-flip tests. For ScienceWorld, the prespecified comparison is source-preserving VSR against Raw Overlay; other contrasts are ablations with unadjusted pairwise tests.

B MEDCALC: FIXED REPRESENTATIONS AND SELECTIVE RELEVANCE RESTORATION

MedCalc separates the benefit of selecting relevant source evidence from the incremental effect of a static contract. The fixed-candidate study evaluates all held-out questions under each available representation. The subsequent mixed-pool analysis asks whether an outcome-blind relevance certificate can preserve already selected inputs and selectively transform Full inputs.

B.1 FIXED-CANDIDATE HELD-OUT EVALUATION

Full supplies the original Skill, Selected supplies task-relevant verbatim source units, and Contract supplies selected evidence together with explicit dependencies, bindings, and checks. Contract Only removes the accompanying source evidence. The development gate selects Selected for 4B, Contract

for 9B, and Full for 27B. Contract verification accepts 279/300 development questions (93.0%) and 759/800 held-out questions (94.9%). The 41 held-out rejections fall back to Full.

Table 9: MedCalc fixed representations on all 800 held-out questions. Entries are correct counts.

Model	Full	Selected	Contract	Contract Only	Dev gate
4B	373/800	512/800	486/800	417/800	512/800
9B	516/800	562/800	558/800	455/800	558/800
27B	631/800	634/800	612/800	529/800	631/800

The gate changes accuracy from 46.6% to 64.0% for 4B, from 64.5% to 69.8% for 9B, and leaves 27B at 78.9%. Across the three models it obtains 1,701/2,400, compared with 1,520/2,400 for Full: a gain of 7.54 percentage points (pp), 293/112 paired wins/losses, exact McNemar $p = 8.93 \times 10^{-20}$, and Skill-cluster 95% CI $[+3.92, +11.38]$ pp. Fixed Selected is slightly higher, at 1,708/2,400. In particular, Selected answers four more 9B questions correctly than the development-selected Contract. Thus a static contract does not consistently improve upon selection. Contract Only reaches 52.1%, 56.9%, and 66.1% for 4B, 9B, and 27B, documenting the cost of removing the evidence.

B.2 Outcome-blind Full/Selected routing

The mixed pool has 40 held-out Skills, each with 20 questions and two input representations, for 1,600 inputs per model. Diagnostics use one representative task and the Skill text, with the median of three local parser repeats. The updated diagnostic run covers 15 development plus 40 held-out Skills, both Full and Selected representations, and three repeats: 330/330 valid diagnostic units, zero length truncations, and zero reasoning characters. Execution answers come from the previously completed 3,300 matched task-condition generations per model (9,900 overall); the routing analysis does not resample answers.

The pre-existing precision criterion, evidence precision below 80%, issues a `distractor_load` certificate for 35 of the 40 held-out Full Skills. Of these 35 triggered candidates, 29 pass the verifier by strictly improving Relevance without reducing obligation coverage. Within this verifier-selected subset of 29 Skills, the median within-Skill Relevance gain is 16.3 points and the minimum gain is 7.5 points; obligation coverage is unchanged for every accepted Skill. Across all paired representations, Selected has higher evidence precision in 31/40 pairs, with a median increase of 22.6 pp. The verifier is idempotent on Selected: all 40/40 such representations are preserved, even where a residual low score remains. The execution gate, fitted only on the development repair subset, enables this repair for 4B and 9B and freezes abstention for 27B.

Across all 40 held-out Skills, Selected increases Relevance in 29 pairs, leaves it unchanged in three, and decreases it in eight. For each Skill, we first take the median Relevance score over the three parser runs separately within the Full and Selected conditions, then compute Selected minus Full. The median of these 40 within-Skill differences is 13.4 points. Both this full-set statistic and the accepted-subset statistic of 16.3 points use the same diagnostic data and aggregation procedure; they differ only in the set of Skills included. In this run, the 29 accepted Skills coincide exactly with the 29 positive-direction pairs.

Table 10: MedCalc selective relevance routing in the 1,600-input mixture. "Always" selects evidence for every input; "Full gain" restricts the effect to the Full half.

Model	No repair	Always	Router	Mixed gain	Full gain	Repaired
4B	55.3%	64.0%	61.9%	+6.6 pp	+13.2 pp	36.2%
9B	67.4%	70.2%	69.6%	+2.2 pp	+4.4 pp	36.2%
27B	79.1%	79.2%	79.1%	+0.0 pp	+0.0 pp	0.0%

Model	Skill-bootstrap 95% CI (pp)	Wins/losses	Exact McNemar p
4B	$[+3.2, +10.4]$	139/33	1.11×10^{-16}
9B	$[+0.2, +4.2]$	71/36	0.000923
27B	$[0.0, 0.0]$	0/0	1

Aggregating the fixed outcomes and paired net wins gives No repair counts of 885/1,600, 1,078/1,600, and 1,265/1,600; Always counts of 1,024/1,600, 1,124/1,600, and 1,268/1,600; and Router counts of 991/1,600, 1,113/1,600, and 1,265/1,600, ordered as 4B/9B/27B.

The mixed-pool effect is smaller than a Full-only contrast because half of the inputs already use Selected. Repairing 29 Full Skills corresponds to 580/1,600 inputs (36.25%, displayed as 36.2%) when the executor gate is enabled. The 27B gate abstains according to its development-frozen decision, although ungated always-selection is slightly higher on held-out data. The fixed-candidate gate in Table 9 and the typed relevance router in Table 10 are different policies with different denominators.

B.3 Scalar diagnostic negative result

An earlier equal-weight scalar experiment uses 15 development Skills parsed three times and 40 held-out Skills parsed once, for 85 diagnostic profiles. Its failure-prediction AUROC is below the token-length baseline for every model. Restoration-headroom AUROC is also close to chance for 4B and 9B. Routing is model-level rather than per-Skill: all 9B Skills are routed to Contract and all 4B Skills are transformed. Accuracy changes for this scalar policy are +17.9, +5.2, and +0.0 pp for 4B/9B/27B, separately from the typed selective-relevance results above.

Table 11: Earlier MedCalc scalar diagnostic experiment. "Headroom" is restoration-headroom AUROC; accuracy columns describe model-level routing.

Model	Failure AUROC: SCS	Failure AUROC: length	Headroom	Full	Router
4B	0.397	0.581	0.514	46.6%	64.5%
9B	0.430	0.599	0.500	64.5%	69.8%
27B	0.470	0.682	0.707	78.9%	78.9%

B.4 Online generation cost and integrity

Table 12: MedCalc mean online cost per generation. All nine rows have zero API errors and zero length truncations.

Model	Representation	Prompt tokens	Completion tokens	Latency (s)
4B	Full	1,571	83	4.99
4B	Selected	1,392	100	5.80
4B	Contract	1,926	95	5.48
9B	Full	1,572	109	5.33
9B	Selected	1,392	105	5.06
9B	Contract	1,926	97	4.90
27B	Full	1,571	124	12.36
27B	Selected	1,392	114	11.31
27B	Contract	1,926	107	11.44

The core MedCalc evaluation contains 9,900 generations with no API errors or length truncations; an additional 1,600 generations implement Contract Only for 4B and 9B. A subsequent 800-question 27B Contract Only evaluation is reported above. Results are append-only and resumable, and the original readout retains the last valid record for each task-condition key.

C ALFWorld: representation, online state, and routing

ALFWorld evaluates whether the executor can maintain action prerequisites, search progress, and completion state over a complete household-task episode. The central fixed-representation comparison uses all 134 `valid_unseen` tasks. A later mixed-pool analysis reuses these matched outcomes to test whether the Progress diagnostic distinguishes a static input from an input that already includes online state.

Table 13: ALFWorld representation ablation on the full 134-task split. All entries are successful environment tasks.

Model	Selected	Static Contract	Contract Only	State Machine
4B	30/134	55/134	51/134	113/134
9B	49/134	52/134	57/134	110/134
27B	109/134	110/134	110/134	129/134

Model	State–Selected (pp)	Wins/losses	Exact McNemar p
4B	+61.9	84/1	4.45×10^{-24}
9B	+45.5	66/5	1.19×10^{-14}
27B	+14.9	20/0	1.91×10^{-6}

C.1 Full-split representation ablation and token control

The static Contract is helpful for 4B, but it is much weaker than the observation-driven State Machine. The latter reaches 84.3%, 82.1%, and 96.3% for 4B, 9B, and 27B. The online controller updates completed nodes from observations, exposes at most three ready steps, and tracks negative search evidence and repeat guards. Token-matched controls repeat source evidence to match the initial Contract input length on each task. They obtain 38/134, 46/134, and 109/134 for 4B, 9B, and 27B, compared with 113/134, 110/134, and 129/134 for State Machine. For 9B, State wins 70 tasks and loses six against this control ($p = 6.31 \times 10^{-15}$). Token matching applies to the initial prompt, not cumulative episode tokens or model calls.

C.2 Static/State mixture and diagnostic response

Each of the 134 tasks appears once with static Selected evidence and once with State Machine, yielding 268 representation inputs per model from the fixed-split paired outcomes. The Progress diagnostic routes all six static task families and preserves all six already-stateful families. It therefore repairs exactly half the inputs. Router and Always State have the same success rates because the latter's intervention is redundant on the already-stateful half.

Table 14: ALFWorld routing on the 268-input Static/State mixture. The discordant tasks and their paired p values are those in Table 13, since only the static half changes.

Model	No repair	Always State	Router	Gain (pp)	Repaired
4B	53.4%	84.3%	84.3%	+31.0	50.0%
9B	59.3%	82.1%	82.1%	+22.8	50.0%
27B	88.8%	96.3%	96.3%	+7.5	50.0%

Adding the paired Selected and State counts gives No repair totals of 143/268, 159/268, and 238/268; reusing State for both representations gives Router totals of 226/268, 220/268, and 258/268 for 4B/9B/27B.

Across six families, Full-to-Selected raises Relevance by 65.6–78.6 points, and Static-to-State raises Progress by 27.8 points in every family. Across 18 family-condition units, the Spearman correlations between SCS and success are 0.715 (4B), 0.769 (9B), and 0.455 (27B). Exact condition-permutation tests stratified by task family give $p = 0.00137$, 0.00103, and 0.0144, respectively. These results show the diagnostic response to the matched interventions.

D Controlled SRA: dependency recovery and diagnostic validity

The controlled SRA experiments hold the knowledge and answers fixed while changing how task-critical relations are expressed. Clean states those relations directly; Dirty introduces aliases, cross-

section producer–consumer references, and implicit state; Recovered attempts to make the same relations explicit again.

D.1 Clean, Dirty, and Recovered execution

Each representation uses seven Skills, 40 questions per Skill, and eight seeds, for 2,240 question-seed outcomes per model and condition. Table 15 gives Skill-level intervals and sign-flip tests, using the Skill as the independent unit of the manipulation.

Table 15: Controlled SRA execution with fixed knowledge. Confidence intervals and sign-flip tests concern Clean minus Dirty.

Model	Clean	Dirty	Recovered
9B	2,031/2,240 (90.7%)	1,846/2,240 (82.4%)	1,908/2,240 (85.2%)
27B	2,069/2,240 (92.4%)	2,013/2,240 (89.9%)	2,042/2,240 (91.2%)

Model	Clean–Dirty (pp)	Skill 95% CI (pp)	Sign-flip p
9B	+8.26	$[+3.75, +13.71]$	0.01562
27B	+2.50	$[+0.31, +5.54]$	0.125

Recovered improves over Dirty by 62 questions for 9B and 29 for 27B, but remains below Clean. The larger model has a smaller Clean–Dirty gap. For 27B, the percentile interval is positive, while the exact Skill sign-flip test is non-significant.

D.2 Equal-token chapter-order control

The ordering control leaves every H2 block byte-for-byte unchanged and randomly permutes whole blocks. Paired prompts have exactly equal token counts. The small reported gaps and intervals crossing zero limit an explanation based only on presentation order.

Table 16: Whole-chapter permutation with identical paired prompt token counts. Differences are Original minus Random order.

Model	Original order	Random order	Difference (pp)	Skill 95% CI (pp)
4B	3,184/3,840	3,180/3,840	+0.10	$[-1.51, +2.03]$
9B	3,334/3,840	3,295/3,840	+1.02	$[-1.12, +2.97]$

D.3 Per-Skill SCS, frozen gold graphs, and repeatability

The local Qwen3.5-9B diagnostic system uses a semantic parser for obligations, bindings, and checkpoints, together with a dependency auditor and deterministic surface checks. Models extract source-backed objects; deterministic code checks references, path depths, and score arithmetic. Table 17 gives the SCS scores for all seven Skills.

Table 17: Controlled SRA diagnostic scores for all seven Skills.

Skill	Clean SCS	Dirty SCS	Clean–Dirty
`logicbench_001`	95.5	70.7	+24.8
`logicbench_005`	85.6	76.0	+9.6
`logicbench_007`	95.5	84.7	+10.8
`logicbench_014`	94.4	70.3	+24.1
`logicbench_015`	84.1	76.0	+8.1
`logicbench_017`	85.6	73.6	+12.1
`logicbench_018`	81.5	0.0	+81.5

Seven Skills, two representations, and three parser seeds produce 42/42 valid diagnostic records. Of 21 Clean/Dirty pairs across repeats, 20 score Clean higher, one ties, and none reverses; the mean gap

is 24.4 points. The mean within-input standard deviation across three parses is 1.18, with median 0.00. All 14/14 representation-specific dependency signatures agree across repeats. A Dirty scalar of zero identifies a diagnostic bottleneck.

Table 18: Two human dependency graphs frozen before automatic parsing. Values are Dependency scores on a 0–100 scale.

Skill	Edges	Clean gold	Clean auto	Dirty gold/auto	Dirty abs. error
`logicbench_001`	3	100.0	100.0	27.8/25.0	2.78
`logicbench_005`	7	100.0	100.0	33.3/33.3	0.00

Across the four Clean/Dirty graph conditions, mean absolute Dependency-score error is 0.69 points. The two frozen graphs assess score agreement with human dependency annotations, complementing the repeated-parser consistency check.

D.4 Clean/Dirty mixed-pool routing

The routing pool contains both Clean and Dirty for every controlled question-seed pair, giving 4,480 inputs per model from the fixed-condition outcomes. A Dependency score below 80 triggers the matching contract repair: all seven Dirty Skills are routed and all seven Clean Skills are preserved. Exactly 50% of inputs are repaired, with no Clean false trigger in this controlled set.

Table 19: SRA routing in the 4,480-input balanced mixture. Intervals and sign-flip tests compare Router with No repair.

Model	No repair	Always VSR	Router	Gain vs No (pp)	Gain vs Always (pp)
9B	86.5%	85.2%	87.9%	+1.38	+2.75
27B	91.1%	91.2%	91.8%	+0.65	+0.60

Model	Repaired inputs	Skill-bootstrap 95% CI (pp)	Sign-flip p
9B	50.0%	$[+0.2, +2.8]$	0.0938
27B	50.0%	$[-0.3, +1.5]$	0.219

Summing the fixed-condition counts gives No repair totals of 3,877/4,480 and 4,082/4,480, Always totals of 3,816/4,480 and 4,084/4,480, and Router totals of 3,939/4,480 and 4,111/4,480 for 9B/27B.

The router avoids applying a recovered representation to already Clean inputs. The 9B interval is positive, but only seven Skills underlie the exact test, whose $p = 0.0938$ does not cross 0.05. The 27B interval crosses zero.

E ScienceWorld: task binding in a frozen demonstration library

ScienceWorld is the independent environment evaluation of task-instance grounding. The library is automatically constructed from training variations and covers 30 official task families; the test uses 332 non-overlapping variations from the 19 families with available variations beyond the offset of 20. All four Qwen conditions are evaluated with native environment outcomes, and errors remain in the denominator. The additional model families are reported in Section O.

E.1 Four-condition confirmatory evaluation

No Repair provides the original Skill. Replacement VSR uses a restored representation in place of the source. Raw Overlay retains the complete library and adds the unbound retrieved demonstration. Source-preserving VSR retains that same library and adds the verified task-bound overlay. Raw Overlay and VSR use the same executor-gate rule, controlling for retrieval, extra material, and the gate while varying whether instance bindings are restored.

Preserving VSR exceeds Raw Overlay at all three scales in the episode-level paired test. Replacement and preserving VSR are close, while preserving VSR retains the input library. On the 227-

Table 20: ScienceWorld success on all 332 test variations under four representation conditions.

Model	No Repair	Replacement VSR	Raw Overlay	Preserving VSR
4B	54/332 (16.3%)	76/332 (22.9%)	48/332 (14.5%)	75/332 (22.6%)
9B	77/332 (23.2%)	103/332 (31.0%)	73/332 (22.0%)	104/332 (31.3%)
27B	125/332 (37.7%)	160/332 (48.2%)	141/332 (42.5%)	166/332 (50.0%)

Model	Preserving VSR–No (pp)	Preserving VSR–Raw (pp)	Paired p, VSR vs Raw
4B	+6.3	+8.1	0.000355
9B	+8.1	+9.3	0.000152
27B	+12.3	+7.5	0.00187

certificate verified subset, 9B gains 14.1 pp over Raw, with a task-family bootstrap interval of $[+5.1, +17.9]$ pp. The 27B family-bootstrap interval against Raw crosses zero, despite the significant episode-level paired result.

E.2 Grounding certificates and threshold sensitivity

The cross-instance diagnostic checks whether a source action exists, the proposed mapping is unique, the target value appears explicitly at the corresponding source position, and the proposed action sequence agrees with deterministic order-preserving substitution. The training task, the current task, and the source actions are read, without test trajectories, answers, or rewards. Parameters that cannot be conservatively aligned remain unresolved and cause abstention.

Scores increase for 291/291 Raw-to-task-bound diagnostic candidates, with 227 accepted certificates at the main threshold. Across 331 task-variant-dimension units diagnosed with three seeds, the threshold-80 decisions agree 98.5% of the time. These analyses use distinct units: 291 candidates for verification coverage, 227 accepted certificates for the verified subset, 331 units for repeatability, and 332 variations for test execution.

Table 21: ScienceWorld Grounding threshold sensitivity. Acceptance counts use the 291 test variations for which the Grounding certificate is applicable. Gains compare threshold-routed VSR with Raw Overlay on the same full set of 332 held-out test variations. Accepted cases use their previously executed VSR-overlay outcomes; all other cases use Raw-overlay fallback.

Threshold	Accepted	Changes vs 80	4B gain (pp)	9B gain (pp)	27B gain (pp)
60	227/291	0	+7.2	+9.6	+7.8
70	227/291	0	+7.2	+9.6	+7.8
80	227/291	0	+7.2	+9.6	+7.8
90	227/291	0	+7.2	+9.6	+7.8
95	223/291	4	+7.2	+9.3	+6.6

This audit recombines previously executed paired Raw-overlay and VSR-overlay outcomes on the same 332 test variations; it does not rerun the models at each threshold. A candidate is accepted when Grounding is applicable, the Raw score is below the threshold, the candidate score reaches the threshold, and the provenance, uniqueness, and closure checks pass. All remaining cases fall back to Raw Overlay. Thresholds 60, 70, 80, and 90 produce the identical set of 227 accepted cases and therefore identical full-test success rates. The corresponding gains over Raw are 7.23, 9.64, and 7.83 pp for Qwen3.5-4B, 9B, and 27B. At thresholds 95 and 100, four cases switch to Raw fallback, leaving 223 accepted cases; the gains remain positive at 7.23, 9.34, and 6.63 pp. At threshold 50, no candidate satisfies all acceptance conditions, so the routed policy reduces exactly to Raw and its gain is zero.

E.3 Development-gate stability and execution cost

All 30 leave-one-family-out recomputations retain Overlay VSR for each model. The lower bootstrap probability for 27B reflects greater sensitivity to family resampling.

Table 22: ScienceWorld development-gate sensitivity to task-family resampling.

Model	Full-development choice	Leave-one-family choice retained	Bootstrap choice probability
4B	Overlay VSR	30/30	83.5%
9B	Overlay VSR	30/30	96.3%
27B	Overlay VSR	30/30	73.7%

Table 23: ScienceWorld online cost per episode, shown as No Repair $\rightarrow$ preserving VSR. Prompt tokens accumulate over the entire episode, not just the initial prompt.

Model	Model calls	Cumulative prompt tokens	Errors	VSR successes
4B	$16.1 \rightarrow 12.1$	$53,483 \rightarrow 43,089$	$8 \rightarrow 8$	75/332
9B	$15.3 \rightarrow 11.1$	$50,151 \rightarrow 35,845$	$6 \rightarrow 6$	104/332
27B	$15.4 \rightarrow 8.9$	$60,347 \rightarrow 38,703$	$1 \rightarrow 1$	166/332

The verified execution gate can directly execute a uniquely legal action and can terminate successful episodes earlier, reducing calls and cumulative prompt tokens despite retaining the library. Offline Grounding certificates are deterministic and add zero generation calls in this study.

F AppWorld: full-denominator system evaluation

AppWorld examines multi-API execution with authentication, runtime identifiers, and repeated side effects. This study evaluates the complete Contract/State system and its fallback policy on both the 343-task accepted subset and the complete 417-task test challenge.

F.1 Development choice, accepted contracts, and fallback

The development comparison uses 143 tasks. The gate selects State Machine for 4B and 9B and Selected Source for 27B. Verification accepts 343/417 contracts (82.3%); for the 74 rejected tasks, the executor discards the generated plan and uses task-selected original API schemas as source fallback.

Table 24: AppWorld development selection and accepted-subset test results. All development cells use 143 tasks, and all accepted test cells use 343 tasks.

Model	Dev Selected	Dev State	Choice	Test Selected	Test Static	Test State
4B	42/143	48/143	State	40/343	70/343	68/343
9B	48/143	61/143	State	43/343	69/343	94/343
27B	88/143	85/143	Selected	148/343	126/343	120/343

Table 25: AppWorld accounting on all 417 test tasks. The complete gate equals accepted-subset gate successes plus measured source-fallback successes.

Model	Accepted gate	Rejected fallback	Complete gate	Complete Selected
4B	68/343	0/74	68/417 (16.3%)	40/417
9B	94/343	2/74	96/417 (23.0%)	45/417
27B	148/343	13/74	161/417 (38.6%)	161/417
All models	310/1,029	15/222	325/1,251	246/1,251

On accepted tasks, fixed Selected obtains 231/1,029, fixed Static 265/1,029, fixed State 282/1,029, and the development gate 310/1,029 (30.1%). Gate versus Selected has 111/32 paired wins/losses and $p = 2.074 \times 10^{-11}$. Source fallback adds $0 + 2 + 13 = 15$ successes, giving the complete-test result of 325/1,251. The complete gate retains a net advantage of 79 successes over complete Selected because both policies use the same fallback on rejected tasks.

F.2 Official ReAct and retrieval baselines

Official ReAct can query the official API documentation during execution. BM25 Skill retrieves and injects up to 20 relevant original API schemas. Both have access to interface definitions and use the complete 417-task denominator.

Table 26: AppWorld complete-test strong baselines.

Model	Official ReAct	BM25 Skill	VSR gate
4B	20/417 (4.8%)	17/417 (4.1%)	68/417 (16.3%)
9B	22/417 (5.3%)	12/417 (2.9%)	96/417 (23.0%)
27B	38/417 (9.1%)	92/417 (22.1%)	161/417 (38.6%)

Model	Wins/losses vs ReAct	p vs ReAct	Wins/losses vs BM25	p vs BM25
4B	55/7	2.43×10^{-10}	59/8	1.02×10^{-10}
9B	78/4	7.61×10^{-19}	87/3	1.96×10^{-22}
27B	126/3	1.05×10^{-33}	88/19	8.48×10^{-12}

The paired comparisons favor the complete VSR gate at every scale, demonstrating system-level gains beyond access to API documentation or lexical retrieval.

F.3 Seven-condition 9B component and token ablation

Table 27: All seven AppWorld 9B component conditions. Prompt tokens are cumulative per task. Different accepted populations are explicitly shown in the success denominator.

Condition	Success	Calls/task	Prompt tokens/task
Selected Source	43/343 (12.5%)	14.5	128,467
Token-matched repeated source	37/343 (10.8%)	15.1	148,186
Unverified relevant schemas	35/391 (9.0%)	18.6	191,093
Unverified linear plan	41/391 (10.5%)	19.6	212,167
Unverified dependency plan	45/391 (11.5%)	19.5	209,591
Verified Contract	69/343 (20.1%)	18.5	174,913
Verified State Machine	94/343 (27.4%)	20.4	209,817

The token-matched baseline repeats original evidence to match the initial Contract prompt on each task. Verified Contract beats this baseline with 46/14 paired wins/losses and $p = 4.22 \times 10^{-5}$. The unverified schema, linear-plan, and dependency-plan sequence increases from 35 to 41 to 45 successes on the same 391-task population; these differences are not significant. This system ablation uses 343-task verified and 391-task unverified populations, so their percentage differences reflect selection as well as representation.

G Verification, gate stability, and compilation cost

We evaluate verification and capability gating through deployment coverage, execution outcomes, and cost. Independent-auditor agreement is examined alongside the risk–coverage tradeoff.

G.1 Nine-point AppWorld risk–coverage readout

The 9B risk–coverage analysis varies the maximum number of compilation rounds and whether auditor-driven repair is permitted. Every full-test accuracy includes source fallback for nonaccepted tasks. The last row removes semantic auditing and checks only structure; its final column describes the newly admitted subset, not all 391 accepted tasks.

Structure-only acceptance adds 48 candidates and raises complete-test successes from 96 to 100. Only 6/48 newly admitted plans succeed; relative to source fallback on this subset, there are 6/2

Table 28: All nine AppWorld 9B risk–coverage settings. "Newly admitted" in the final row means the 48 candidates rejected by the default semantic verifier but admitted by structure-only checks.

Max rounds	Auditor repair	Coverage	All 417 accuracy	Accepted accuracy
1	No	20/417 (4.8%)	51/417 (12.2%)	15/20 (75.0%)
1	Yes	60/417 (14.4%)	57/417 (13.7%)	27/60 (45.0%)
2	No	39/417 (9.4%)	52/417 (12.5%)	19/39 (48.7%)
2	Yes	174/417 (41.7%)	80/417 (19.2%)	60/174 (34.5%)
3	No	45/417 (10.8%)	51/417 (12.2%)	21/45 (46.7%)
3	Yes	248/417 (59.5%)	82/417 (19.7%)	71/248 (28.6%)
Unlimited	No	47/417 (11.3%)	51/417 (12.2%)	22/47 (46.8%)
Unlimited	Yes (default)	343/417 (82.3%)	96/417 (23.0%)	94/343 (27.4%)
Structure only	Yes	391/417 (93.8%)	100/417 (24.0%)	6/48 newly admitted

wins/losses and $p = 0.289$, a non-significant improvement. The high 15/20 success rate at the strictest setting comes with low coverage, leaving most tasks to fallback.

G.2 Development-gate bootstrap and held-out oracle

The gate-stability readout resamples development groups and recomputes the representation choice. The held-out oracle is shown only retrospectively and never selects the deployed policy.

Table 29: MedCalc/AppWorld gate stability. Bootstrap gives the probability of retaining the full-development choice; leave-one-group-out gives its retention rate.

Dataset	Model	Dev choice	Bootstrap	Leave-one-group	Held-out oracle
MedCalc	4B	Selected	55.6%	86.7%	Selected
MedCalc	9B	Contract	97.6%	100.0%	Selected
MedCalc	27B	Full	71.3%	100.0%	Selected
AppWorld	4B	State	76.2%	100.0%	State
AppWorld	9B	State	93.9%	100.0%	State
AppWorld	27B	Selected	66.4%	100.0%	Selected

MedCalc 4B retains its choice in only 55.6% of grouped bootstrap draws, consistent with close candidate scores. AppWorld 9B is more stable at 93.9%. MedCalc's 9B/27B development choices differ from the retrospective Selected oracles. ScienceWorld's separate 30-family analysis appears in Table 22.

G.3 Compilation cost and seven-point amortization

Table 30: Offline compiler cost. Attempts, calls, prompt tokens, and completion tokens are means over the corresponding compilation collection.

Collection	Accepted	Attempts	Calls	Prompt tokens	Completion tokens
MedCalc development	279/300 (93.0%)	1.27	2.92	8,143	1,435
MedCalc held-out	759/800 (94.9%)	1.23	2.98	8,213	1,470
AppWorld test challenge	343/417 (82.3%)	3.52	9.32	146,227	8,238

AppWorld's high compilation cost arises mainly from auditing large collections of API schemas. The amortization analysis measures the offline charge assigned to repeated executions of the same Contract, using a baseline of 164,626 tokens. This contract-reuse cost is distinct from the corpus-level prompt and completion means in Table 30.

Reuse reduces the offline charge, while the online premium remains approximately 82,739 tokens per task relative to Selected. AppWorld therefore retains an execution-cost premium even with repeated compilation reuse, in contrast to ScienceWorld's lower cumulative episode cost.

Table 31: AppWorld Contract-reuse amortization. The last column is the fixed State-minus-Selected accuracy difference associated with this cost analysis.

Reuse count	Offline tokens/execution	Total extra tokens vs Selected	State–Selected (pp)
1×	164,626	247,364	+14.9
2×	82,313	165,051	+14.9
5×	32,925	115,664	+14.9
10×	16,463	99,201	+14.9
20×	8,231	90,970	+14.9
50×	3,293	86,031	+14.9
100×	1,646	84,385	+14.9

G.4 Typed dispatch checks and independent-auditor conflict

The shared dispatcher audit contains 269 verified certificates: 29 Relevance, seven Dependency, 227 Grounding, and six Progress. Each certificate is sent to its matching compiler pass and to all three wrong passes. The 269/269 matched cases pass the shared kernel; all 807/807 mismatched cases are rejected with `typed_pass_mismatch`. This deterministic check validates the interface and type routing over the existing certificates.

Table 32: Composition of the typed-dispatch audit. Each certificate has one matched pass and three wrong passes. All 807 wrong-type pairings are rejected.

Certificate type	Certificates	Matched pairs	Wrong-type pairs
Relevance	29	29	87
Dependency	7	7	21
Grounding	227	227	681
Progress	6	6	18
Total	269	269	807

An independent local Qwen3.5-27B auditor agrees with 134/134 accepted ALFWorld contracts, with zero parse errors. On AppWorld, the same independent auditor rejects all 100/100 sampled contracts previously accepted by the original verifier and agrees with all 74/74 original rejections.

H Natural-Skill external validity and negative routing results

The controlled mixtures make the intended defect boundary unusually clear. The natural-Skill studies test whether such representation preferences can be discovered without a known Clean/Dirty or Static/State construction. They do not show a reliable benefit from applying a complete overlay whenever an equal-weight score is low.

H.1 Nineteen native SRA Skills

All 19 native SRA-Bench Skills are treated as unknown inputs. One task per Skill is reserved for three SCS diagnostic repeats before threshold freezing. The compiler reads only the Skill, without questions or answers. The remaining 39 questions per Skill, evaluated with eight sampling seeds, form 5,928 held-out paired outcomes. Nine Skills are routed and ten are preserved; 17 candidates pass verification and two strictly fall back to Source.

The short replacement loses 2.55 pp. Fixed preserving Overlay gains 0.29 pp, with Skill-cluster 95% CI $[-1.10, +1.79]$ pp. Frozen SCS routing obtains 87.94%, a change of -0.19 pp with CI $[-0.59, +0.15]$ and exact Skill sign-flip $p = 0.406$. The retrospective oracle reaches 89.29%, indicating representation heterogeneity that this router does not predict successfully.

Two plausible failure mechanisms are a mismatch between online-state restoration and one-shot logic Skills, and the removal of worked examples needed to interpret opaque labels by short replacement.

Table 33: Natural SRA held-out representation and routing results. The per-Skill oracle is retrospective and is not a deployable policy.

Policy	Correct	Accuracy	Change vs Original (pp)
Original native Skill	5,224/5,928	88.1%	+0.0
Fixed short VSR, replacing Source	5,073/5,928	85.6%	−2.5
Fixed source-preserving Overlay	5,241/5,928	88.4%	+0.3
Frozen SCS-to-Overlay router	5,213/5,928	87.9%	−0.2
Retrospective per-Skill oracle	5,293/5,928	89.3%	+1.2

H.2 CHAMP AND THEOREMQA

The frozen held-out study contains 98 benchmark-provided Skills and 2,144 paired seed samples. Overall accuracy is 40.95% for Original, 40.90% for a typed overlay on every input, and 40.86% for SCS routing, with Skill-cluster intervals crossing zero. Table 34 gives the dataset-level results.

Table 34: Natural CHAMP/TheoremQA held-out accuracy.

Collection	Original	Always Typed VSR	SCS routing
Overall	41.0%	40.9%	40.9%
CHAMP	40.6%	40.9%	40.1%
TheoremQA	41.3%	40.9%	41.5%

The dataset difference is consistent with different evidence requirements: TheoremQA often resembles local formula retrieval, whereas CHAMP can depend on a fuller composition of reasoning knowledge. A separate development audit of 23 Skills automatically extracted from demonstrations contains 628 paired samples and obtains 21.97% both before and after restoration. These Skills contain missing or incorrect knowledge; organizing available evidence cannot supply absent rules. Adding an answer-blind closure gate yields only approximately $+0.05$ pp on the development natural-Skill evaluation, still non-significant.

I VISUAL SKILL NEGATIVE RESULT ON THE COMPLETE ALFWORLD SPLIT

The visual experiment tests an information-preserving representation change. It compiles one generic ALFWorld Skill into a structured overview and a diagram, then supplies those views in addition to the original text. The full evaluation does not establish a visual benefit.

I.1 MODEL, COMPILATION ARTIFACT, AND ACTION INTERFACE

The experiment uses local Qwen3-VL-8B-Instruct, temperature zero, seed 41, `max_tokens`=256, and a 50-step episode limit. Eight independent vLLM services on eight NVIDIA H20 GPUs distribute tasks across 32 ALFWorld environment processes. All 134 `valid_unseen` tasks are run under all four conditions, for 536 episodes and 18,905 model calls, with no HTTP or environment errors.

The original Skill has five generic principles: parse target objects and transformations; choose only admissible commands; systematically explore and remember locations; satisfy prerequisites before placing objects; and stop only when the environment reports success. It supplies no room layout and no separate heat, cool, clean, two-object, or lamp-task templates. The native pipeline converts this text into semantic units and a dependency hierarchy, a validated manifest, and matching structured-text and PNG views. Both Structure and Diagram retain the complete original Skill; only the extra overview's modality differs.

Each episode resets the same game and invokes the repository's native `make_messages()` once. Each environment turn appends the current observation, a summary of the latest 12 steps, and numbered admissible commands. The model emits a short decision and `<action_index>N</action_index>`; the runner maps the index to a legal command. The

Table 35: The four visual-study conditions use the same tasks and native condition constructor.

Condition	Initial context
No Skill	Task description only
Text	Task plus original `SKILL.md`
Text + Structure	Task, original Skill, and compiled structured overview
Text + Diagram	Task, original Skill, and compiled diagram PNG

initial Skill or image remains available while the full conversation retains the latest eight turns to stay within a 16k context limit.

A fresh local-Qwen compilation failed strict manifest validation, so this evaluation reused previously successful, schema- and coverage-audited artifacts generated from the identical byte-level Skill source. All 536 environment episodes were newly run with these artifacts.

I.2 Overall rates, uncertainty, and task-family counts

Table 36: Complete visual-study results. Parse-error counts are episodes containing at least one parsing failure, not numbers of failed calls.

Condition	Success	Rate	95% Wilson CI	Mean steps	Parse-error episodes
No Skill	60/134	44.8%	36.6–53.2%	34.8	11
Text	59/134	44.0%	35.9–52.5%	34.5	4
Text + Structure	57/134	42.5%	34.5–51.0%	35.1	10
Text + Diagram	50/134	37.3%	29.6–45.7%	36.6	12

Table 37: Visual-study successes in every ALFWorld task family. Column totals reproduce Table 36.

Task family	N	No Skill	Text	Structure	Diagram
Find and inspect under a lamp	18	6	4	5	4
Pick and place one object	24	24	24	23	23
Clean then place	31	18	19	16	13
Cool then place	21	2	2	2	1
Heat then place	23	2	3	3	3
Pick and place two objects	17	8	7	8	6
Total	134	60	59	57	50

Simple placement is near ceiling, while heating and cooling are near floor. Cleaning is a more discriminative family in this run, and Diagram also trails Text there. These distributions limit how much a generic representational overview can improve overall success.

I.3 All paired comparisons and reliability checks

Table 38: Visual-study paired tests on the same 134 tasks. Wins/losses are unique successes for the first/second condition.

Comparison	Wins	Losses	Two-sided exact McNemar p
Text vs No Skill	13	14	1.000
Structure vs Text	14	16	0.856
Diagram vs Text	11	20	0.150
Diagram vs Structure	10	17	0.248
Diagram vs No Skill	8	18	0.076

No paired comparison reaches $p < 0.05$. Diagram's point estimate is 5.2 pp below Structure, and its paired counts favor the nonvisual conditions, with no statistically significant difference.

All 134 tasks have complete four-condition pairing. Of 18,905 calls, 18,863 end with `stop`; all 42 `length` terminations occur in a single Text episode. The total `reasoning_content` character count is zero. Restricting to the 106 tasks with no parse failure in any condition gives success counts 55, 50, 50, and 45 for No Skill, Text, Structure, and Diagram, respectively, leaving Diagram lowest.

The result is consistent with three explanations: the generic Skill contributes little task-specific knowledge, visual reformatting adds no new executable rules, and small-model action loops persist despite an overview.

J TWELVE-CONDITION SURFACE-PERTURBATION DOSE STUDY

The surface-dose experiment tests whether simply adding more apparent disorder produces a monotonic execution penalty. It uses a Selected baseline of 49/134 and four perturbation families at three severity levels, for 12 conditions and 1,608 episodes. All environment runs complete without infrastructure errors.

Table 39: All surface-perturbation conditions. Effects and paired p values compare each condition with the 49/134 unperturbed Selected baseline.

Perturbation family	Severity	Success	Change (pp)	Paired p
Irrelevant content	Mild	49/134	+0.0	1.000
Irrelevant content	Medium	49/134	+0.0	1.000
Irrelevant content	Severe	46/134	−2.2	0.629
Remote dependencies	Mild	41/134	−6.0	0.115
Remote dependencies	Medium	48/134	−0.7	1.000
Remote dependencies	Severe	48/134	−0.7	1.000
Variable aliases	Mild	43/134	−4.5	0.238
Variable aliases	Medium	46/134	−2.2	0.581
Variable aliases	Severe	45/134	−3.0	0.481
Implicit state	Mild	49/134	+0.0	1.000
Implicit state	Medium	45/134	−3.0	0.388
Implicit state	Severe	45/134	−3.0	0.481

No family shows a statistically established monotonic decline from Mild through Severe. Moving a few steps into a registry, adding irrelevant sentences, or introducing a few aliases is therefore insufficient to define a general execution-difficulty scale.

K EXECUTION CASES AND EXPLANATORY EXAMPLES

The cases show interventions at the level of source units, bindings, and actions. ScienceWorld provides a test episode, MedCalc a held-out Skill and its 20-question outcomes, and ALFWorld an alarm-clock execution excerpt. The AppWorld case pairs an observed baseline failure with an explanatory dependency contract and idealized SCS arithmetic. The pepper-shaker and CKD-EPI constructions are illustrative method examples.

K.1 SCIENCEWORLD: AN ANIMAL, A YELLOW BOX, AND THE BATHROOM

The current task asks: "Your task is to find a(n) animal. First, focus on the thing. Then, move it to the yellow box in the bathroom." The retrieved training demonstration instead ends at the red box in the kitchen. A Raw overlay retains those stale training-instance destinations. The verified overlay preserves the complete library and adds only the task-text-supported substitutions red-to-yellow and kitchen-to-bathroom. Animal identity has no reliable static alignment and is left for observations to resolve.

For the changed slots, both Raw and VSR pass source-step existence and unique-mapping checks. Raw fails explicit target-value occurrence and agreement with the deterministic replacement; VSR passes both. The resulting slot Grounding scores are 50 and 100. The score change concerns the destination mapping, while animal identity is resolved during execution.

Table 40: ScienceWorld source action sequences in the animal case. Object phrases are reproduced verbatim.

Step	Raw retrieved source	Task-bound overlay
1	`open door to kitchen`	`open door to bathroom`
2	`go to kitchen`	`go to bathroom`
3	`open door to outside`	`open door to outside`
4	`go to outside`	`go to outside`
5	`look around`	`look around`
6	`focus on blue jay`	`focus on blue jay`
7	`pick up blue jay`	`pick up blue jay`
8	`open door to kitchen`	`open door to bathroom`
9	`go to kitchen`	`go to bathroom`
10	`move egg blue jay egg in inventory to red box`	`move egg blue jay egg in inventory to yellow box`

Table 41: ScienceWorld 9B execution, `find-animal__v250`. Raw finishes at score -100; VSR finishes at 100. Empty VSR cells indicate that its nine-action trajectory has already ended.

Step	Raw execution	VSR execution
1	`teleport to kitchen`	`go to bathroom`
2	`go to outside`	`teleport to outside`
3	`look around`	`look around`
4	`look around`	`look around`
5	`look around`	`look around`
6	`look around`	`focus on common toad`
7	`look around`	`pick up common toad`
8	`look around`	`teleport to bathroom`
9	`look around`	`move common toad to yellow box`
10	`move egg giant tortoise to kitchen`	
11	`teleport to bathroom`	
12	`look around`	
13	`teleport to outside`	
14	`teleport to bathroom`	
15	`look around`	
16	`look around`	
17	`look around`	
18	`look around`	
19	`teleport to outside`	
20	`teleport to bathroom`	
21	`focus on orange`	

The test scores for this episode are No Repair $= -100$, Raw Overlay $= -100$, and VSR $= 100$. The VSR execution resolves the animal to a common toad from the environment and moves it to the requested yellow box. The destination substitutions are supported by task text, while the entity comes from observation.

K.2 MEDCALC: UNIT DEPENDENCIES IN A CKD-EPI EXAMPLE

This illustrative example shows unit conversion and binding dependencies in a one-shot task. For a 61-year-old male whose creatinine is 676 μmol/L, the contract first binds age, sex, and the measurement's unit. It then obtains creatinine in mg/dL through the source-specified conversion $676/88.42$ before selecting the source formula's male branch with $\mathrm{Scr} > 0.9$. The dependency order prevents the formula from consuming the unconverted measurement, and the completion check verifies the expected eGFR unit, mL/min/1.73 m^2.

K.3 MEDCALC: SELECTING THE EVIDENCE FOR FRIEDEWALD LDL

The held-out task requests LDL from admission lipid measurements using the Friedewald formula. The Skill has six source units: an introduction, required inputs, the formula, unit conversion, a calculation tool, and another patient's worked example. The current measurements are already in mg/dL. Without seeing an answer, the parser identifies `u1/u2/u3/u5` as sufficient evidence, while the unused conversion branch `u4` and other-patient example `u6` are irrelevant to this instance.

Full has obligation coverage 1.00, evidence precision 0.611, and Relevance 75.9, producing a `distractor_load` certificate. The candidate retains the selected units verbatim, keeps coverage at 1.00, and raises precision and Relevance to 1.00 and 100. Across this held-out Skill's 20 questions in the MedCalc evaluation, 4B improves from 1/20 to 18/20 and 9B from 12/20 to 18/20.

K.4 APPWORLD: MOVE EVERY WATCH FROM WISH LIST TO CART

The task is to move all watches from the Amazon wish list to the cart. In the observed baseline failure, Official ReAct first invokes the nonexistent `add_to_cart`, receives an exception, and then queries API listings. The illustrative VSR contract specifies valid APIs and parameter providers, with passwords, tokens, and product identifiers obtained at runtime.

Table 42: Illustrative AppWorld watch contract. Runtime outputs become available through the corresponding API responses.

Step	Required operation	Produced value or completion evidence
1	`show_account_passwords`	Account credentials
2	`amazon.login`, using Step 1	`access_token`
3	`amazon.show_wish_list`, using Step 2	Items including `product_id`, name, and quantity
4	For every item whose name contains "watch", call `move_product_from_wish_list_to_cart`	Actual product ID, quantity, and access token from earlier responses; success of each move
5	`supervisor.complete_task` after all matching moves succeed	The entire "all watches" obligation is satisfied

An idealized numerical example illustrates SCS computation for this task. Of 1,470 source tokens, 470 are relevant, with complete obligation coverage; two required dependency paths have depths three and two; one runtime parameter has two candidate providers and another has none; per-item success and final completion checks are absent. Table 43 gives the resulting scores.

K.5 ALFWORLD: TWO PEPPER SHAKERS IN A DRAWER

This illustrative six-step state machine makes progress explicit for "Put two pepper shakers in the drawer." The steps are: (1) locate two distinct pepper shakers; (2) take the first and require an observation establishing that it is held; (3) place it in the drawer and require placement evidence; (4) take the second, only after the first placement is established; (5) place the second in the drawer; and

Table 43: Illustrative watch-task scoring example. The perfect restored scores are an idealized construction, not measured AppWorld benchmark scores.

Dimension	Illustrative original	Idealized restored	Change
Relevance	48.5	100.0	+51.5
Dependency	41.7	100.0	+58.3
Grounding	25.0	100.0	+75.0
Progress	30.0	100.0	+70.0
Geometric SCS	35.1	100.0	+64.9

(6) terminate only when observations establish that both objects are in the drawer. The dependency $s_4 \leftarrow \{s_1, s_3\}$ records the distinct-object requirement and completed first placement. Each step's product and success check expose progress without requiring the executor to infer from a static two-object schema whether the first object has already been handled.

K.6 ALFWorld: inspect an alarm clock under a desk lamp

For "look at alarmclock under the desklamp," the four-step execution excerpt in Table 44 shows how a frontier records completed work and exposes the next prerequisite. It illustrates the runtime interface used in the full 134-task comparison.

Table 44: ALFWorld alarm-clock execution excerpt and state updates.

Step	Action	State or observation
1	`look`	Locate alarm clock is NEXT.
2	`go to desk 1`	Observation establishes the location.
3	`take alarmclock 1 from desk 1`	Locate and take/approach are DONE; locate lamp is NEXT.
4	`use desklamp 1`	The environment reports task success.

The state representation records which prerequisites have been observed and which action is ready. It avoids relying solely on the model's recollection of a static instruction such as "find the object, then find and use the lamp." This excerpt illustrates the state interface associated with the 9B Selected-to-State change from 49/134 to 110/134.

L Diagnostic definitions and algorithmic interfaces

This section defines the source-linked diagnostic scores, development-time deployment decisions, and online execution interface.

L.1 Source-linked graph and applicable dimensions

The diagnostic graph is $G = (U, O, E, B, Q, z)$. Source units U retain their text spans, O contains task obligations, E contains required producer–consumer dependencies, B contains parameters or instance slots, Q contains execution steps, and z describes completion. A semantic extraction is admissible only when its supporting span exists in the Skill, task, or currently available observation. Each span citation makes the extraction traceable to its source.

The profile contains four scores in $[0, 1]$ or N/A. Let K be the set of applicable dimensions. An empty set of dependencies, parameters, or interactive progress obligations yields N/A for that dimension, not 1. This treatment matters in natural one-shot reasoning Skills, whose lack of online state is not a defect requiring a state machine. Weighted averages below assume nonnegative weights with positive total mass over the applicable items.

L.2 RELEVANCE

For each extracted obligation o_j, let A_j be its recorded source support. Let $t_i > 0$ be the token mass of unit u_i and $R \subseteq U$ the relevant or obligation-cited units. For nonempty O and U, the score combines obligation coverage with relevant evidence precision:

$$\text{cov} = \frac{|\{j : A_j \neq \varnothing\}|}{|O|}, \qquad \text{prec} = \frac{\sum_{u_i \in R} t_i}{\sum_{u_i \in U} t_i}, \tag{8}$$

$$C_{\text{rel}} = \begin{cases} 0, & \text{cov prec} = 0, \\ 2\text{cov prec}/(\text{cov} + \text{prec}), & \text{otherwise.} \end{cases} \tag{9}$$

Coverage comparisons require a common obligation set O. The relevance certificate additionally requires precision below 0.8, and acceptance requires a higher target score with no decrease in aggregate coverage. Aggregate and pointwise coverage differ: covering only o_1 before repair and only o_2 afterward gives the same coverage of $1/2$. Pointwise preservation requires checking each previously covered obligation.

L.3 DEPENDENCY

For edge e, let h_e be its effective recovery depth. A direct source-step relation has depth one; each alias, pronoun, cross-section reference, or implicit-state resolution adds one. With edge weight a_e,

$$r_e = \begin{cases} 1/h_e, & e \text{ uniquely resolved}, \\ 0, & \text{otherwise}, \end{cases} \qquad C_{\text{dep}} = \frac{\sum_e a_e r_e}{\sum_e a_e}. \tag{10}$$

The score is a deterministic description of the extracted dependency graph. Its inverse-depth term is not an empirically calibrated probability of recovering an edge.

L.4 GROUNDING AND SPECIALIZED SLOT CERTIFICATES

For required parameter b, retain providers with explicit, source-grounded values in the evidence available at scoring time, and deduplicate their canonical values into V_b. Multiple references to the same value do not create ambiguity. The value-based score is

$$g_b = \begin{cases} 0, & |V_b| = 0, \\ 1/|V_b|, & |V_b| > 0, \end{cases} \qquad C_{\text{grd}} = \frac{\sum_b w_b g_b}{\sum_b w_b}. \tag{11}$$

This statistic describes explicit value availability and ambiguity at scoring time. An unresolved value may have a supported dependency on a future observation or tool response. Runtime closure checks whether this dependency is specified; actual binding awaits observation. Thus an empty V_b can coexist with a valid procedure for obtaining the value.

ScienceWorld's cross-instance analysis uses a narrower slot certificate. For changed slot s, P_s checks that source actions exist, U_s checks uniqueness, T_s checks that the task value is explicit at the corresponding location, and A_s checks agreement with order-preserving deterministic substitution:

$$c_s = 25(P_s + U_s + T_s + A_s), \qquad C_{\text{slot}} = \frac{1}{|B_\Delta|} \sum_{s \in B_\Delta} c_s. \tag{12}$$

If no slot changes, this certificate is N/A; an unalignable changed instance scores zero and abstains. The animal case's 50-to-100 change is measured by this specialized slot certificate, separately from generic Grounding and the scalar SCS.

L.5 PROGRESS AND SCALAR SUMMARY

Let C_{step} measure weighted observable postconditions, C_{stop} observable completion, C_{online} observation-based frontier updates, and C_{guard} repeated-action guards when relevant. With applicability flag I_G,

$$C_{\text{prog}} = \frac{0.40C_{\text{step}} + 0.25C_{\text{stop}} + 0.25C_{\text{online}} + 0.10I_G C_{\text{guard}}}{0.90 + 0.10I_G}. \tag{13}$$

Manifests without the online-state field use the legacy 0.60/0.30/0.10 weighting. Inapplicable components are removed and weights renormalized.

The optional scalar is a weighted geometric mean over applicable dimensions,

$$\mathrm{SCS}_{\mathrm{scalar}} = 100 \exp\left(\frac{\sum_{k\in K} w_{M,k} \log C_k}{\sum_{k\in K} w_{M,k}}\right). \tag{14}$$

Only positively weighted applicable dimensions enter the sum; a zero score on such a dimension gives a zero scalar by continuous extension. With no positively weighted applicable dimension, the scalar is undefined. The scalar summarizes the profile; the full profile and typed certificate guide diagnosis and repair.

L.6 Frozen development decisions

Algorithm 2 Development-only representation selection and type enablement

Require: Development tasks D_{dev}, executor M, eligible representations $\mathcal{A}$
for $a \in \mathcal{A}$ **do**
  $v_a \leftarrow \mathrm{Evaluate}(a, M, D_{\mathrm{dev}})$ ▷ same evaluator, source fallback included
end for
$a^*_M \leftarrow \mathrm{argmax}_{a\in\mathcal{A}}\, v_a$
Freeze a^*_M before held-out evaluation
for target defect type k **do**
  $D_k \leftarrow \{u \in D_{\mathrm{dev}} : \mathrm{Certificate}_k(u) \text{ is active and verified}\}$
  $\delta_{M,k} \leftarrow \mathrm{MeanPairedGain}(\mathrm{repair}_k, \mathrm{source}; M, D_k)$
  $\gamma_{M,k} \leftarrow \mathbf{1}[\delta_{M,k} > 0]$
end for
Freeze γ and the diagnostic thresholds; use no test outcome for updating them

Algorithm 2 presents two distinct deployment protocols: Full/Selected/Contract/State representation selection and type-specific capability gating. Each protocol freezes its decision on development data. In the absence of supporting development evidence, the interface leaves the source unchanged.

Bootstrap and leave-one-group-out analyses assess the stability of these development choices; held-out evaluation measures their execution benefit.

L.7 Online execution interface

Algorithm 3 Observation-driven execution frontier

Require: Verified contract C, task x, executor M, interaction budget H
Initialize unresolved bindings and unconfirmed steps; observe o_0
for $t = 0, \ldots, H-1$ **do**
  Resolve only bindings supported by o_t or earlier actual responses
  Confirm postconditions supported by observations; update negative search evidence
  if the completion predicate is observed **then**
    return observed completion
  end if
  $F_t \leftarrow \mathrm{ReadySteps}(C, \text{confirmed prerequisites})$
  Expose at most three ready steps and the relevant source evidence
  $a_t \leftarrow M(x, o_{\le t}, F_t, \text{bindings})$
  Check action legality, repeated side effects, and unchanged-state guards
  Execute a legal action consistent with the guard policy; obtain o_{t+1}
end for
return unresolved outcome at the budget limit

Algorithm 3 maintains the execution frontier through observation-supported bindings, confirmed postconditions, and action guards. Completion is determined from observations.

M EXPLANATORY RECOVERY MODEL: ASSUMPTIONS AND PROOFS

The model explains how available knowledge can remain difficult to execute, identifies sufficient conditions for a beneficial intervention, and delineates their scope. The diagnostic scores above are operational design choices; their formulas and weights are not derived from the recovery probabilities below.

M.1 CHAIN-RULE DECOMPOSITION

Condition throughout on a fixed Skill, task, and executor (S, x, M). Let A be successful execution, fix an ordered set of recovery events $E_1, \ldots, E_m$, and write $E = \cap_j E_j$. The essential assumption is $A \subseteq E$: every successful execution in the specified problem requires these relations. When $P(E) > 0$, all prefixes have positive probability, so $q_j = P(E_j \mid E_{<j})$ and $\kappa = P(A \mid E)$ are defined. Since $P(A \cap E) = P(A)$, the chain rule gives

$$X_M(S, x) = P(A) = \kappa P(E) = \kappa \prod_{j=1}^{m} q_j. \tag{15}$$

No independence assumption is used. If $P(E) = 0$, the subset assumption directly gives $P(A) = 0$; we do not define κ or later conditional probabilities by conditioning on a null event. For $P(E) > 0$, all $q_j > 0$, and $B = -\sum_j \log q_j = -\log P(E)$ gives $X = \kappa e^{-B}$. Changing the event order changes individual conditional probabilities, but not their product or B.

Fixing a plan alone does not make its relations necessary for all successful behavior. If alternative successful plans bypass E, then

$$P(A) = P(A \cap E) + P(A \cap E^c), \tag{16}$$

and the simple recovery model omits the second term. Its scope is therefore a problem restricted to the specified plan, or a set of relations necessary across all admissible successful plans. The relation set, its order, and this necessity assumption must remain the same when comparing representations.

M.2 BOUNDED-INSPECTION SEPARATION

Fix integers $1 \le k \le b$, and let $N \ge b$. The construction contains N true source units, of which a specified k are necessary for the chosen task. An executor samples a uniformly distributed b-element subset without replacement and cannot solve the task without all k units. There are $\binom{N}{b}$ possible inspected sets, and exactly $\binom{N-k}{b-k}$ contain all required units. Thus its probability of full recovery is

$$P(E) = \frac{\binom{N-k}{b-k}}{\binom{N}{b}} = \prod_{i=0}^{k-1} \frac{b-i}{N-i} \quad (k \le b \le N). \tag{17}$$

For fixed b, k, this tends to zero with N. Because $0 \le X_M(S_N, x) \le P(E)$, success also tends to zero even if residual competence varies with N. If $b < k$, recovery and success are zero.

The added units leave task-relevant knowledge unchanged while increasing total content and length. The construction isolates evidence-access difficulty under a fixed inspection budget: semantic correctness and task sufficiency can coexist with vanishing execution success.

M.3 SUFFICIENT CONDITIONS FOR RESTORATION BENEFIT

Compare the original and intervened representations using the same necessary events and order, with positive $P(E)$ in both conditions. If $q'_j \ge q_j$ for every j and $\kappa' \ge \kappa$, multiplication of nonnegative factors yields

$$X' = \kappa' \prod_j q'_j \ge \kappa \prod_j q_j = X. \tag{18}$$

When $X > 0$, division gives $X'/X = (\kappa'/\kappa)\exp(B - B')$. If the baseline recovery probability or κ is zero, then $X = 0$ and non-decrease follows from $X' \ge 0$ without using a ratio. These are sufficient, not necessary, conditions: some recovery probabilities may decrease while the total product improves. Conversely, improved recovery can be outweighed by lower residual competence; for one event, changing (q, κ) from $(0.8, 0.9)$ to $(0.9, 0.5)$ lowers success from 0.72 to 0.45.

M.4 WHY MARGINAL RECOVERY INCREASES ARE INSUFFICIENT

An increase in every marginal recovery probability does not by itself prove an increase in their conjunction. For two events, consider a baseline with $P(E_1 \cap E_2) = 0.6$ and $P(E_1^c \cap E_2^c) = 0.4$. Both marginals are 0.6. An alternative with joint probabilities $P(E_1 \cap E_2) = 0.4$, $P(E_1 \cap E_2^c) = 0.3$, and $P(E_1^c \cap E_2) = 0.3$ has marginals 0.7 but a lower conjunction probability. The conditional probability $P(E_2 \mid E_1)$ drops from one to $4/7$.

M.5 VERIFICATION AND ABSTENTION

Type matching routes a certificate to its designated repair pass; source-span checks establish traceability; coverage checks constrain the proportion of extracted obligations with recorded support; closure checks assess prerequisites in the extracted structure. These checks operate on the parser's representation.

An overlay preserves the complete source text in the execution input, provided no truncation occurs. Evidence selection preserves selected passages verbatim and applies score and coverage checks.

Exact abstention preserves the baseline execution distribution when the input, policy, budget, and relevant environment state are unchanged. Diagnostic computation is accounted for separately as overhead.

The empirical studies examine these mechanisms through distinct interventions. Controlled SRA tests execution under retained task knowledge and altered dependencies; chapter reordering supplies a token-matched control; ALFWorld evaluates an online execution scaffold with state updates and guards; ScienceWorld compares task-bound and raw retrieved overlays under the same gate. Each study measures how the corresponding intervention changes diagnostic responses and task outcomes.

N FULL-RESOLUTION STATISTICAL FIGURES

These figures compare experimental configurations and cost-amortization scenarios. Curves connect the evaluated settings, and the captions specify the units and sample sizes.

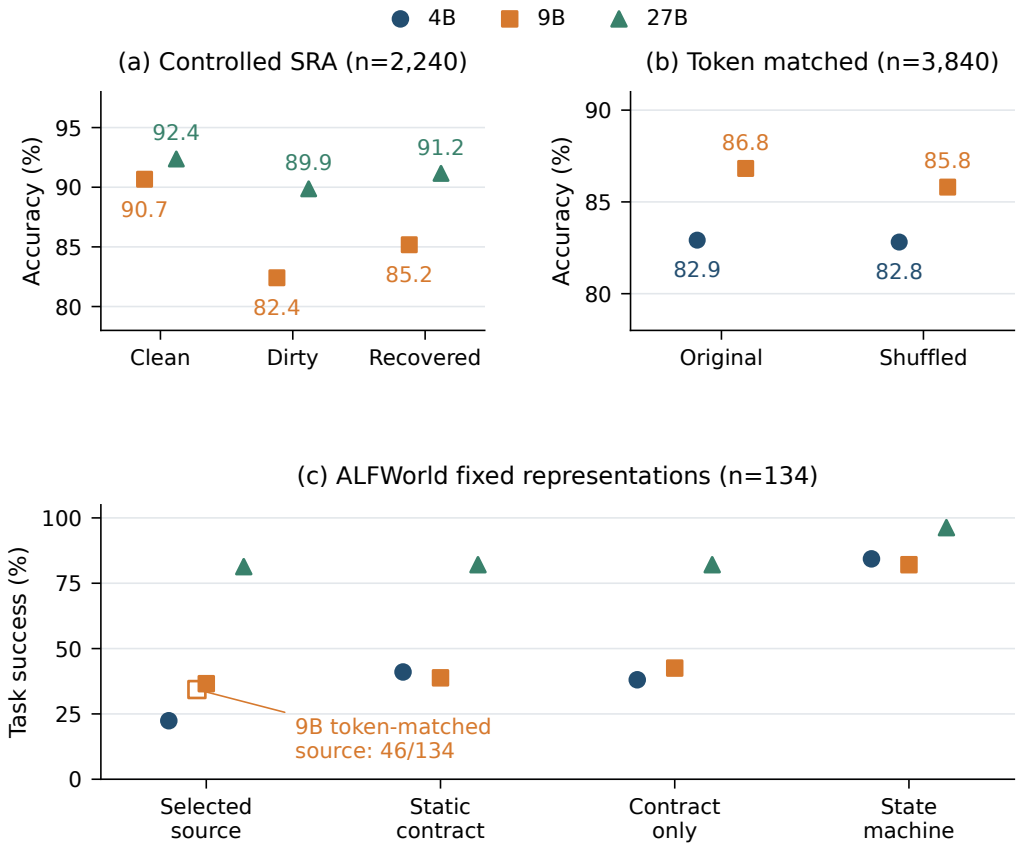


Figure 5: Full controlled representation comparisons. SRA uses 2,240 task-seed observations per condition, whereas the separate chapter-order control uses 3,840. ALFWorld uses 134 tasks per fixed representation; its 9B token-matched source point is shown separately.

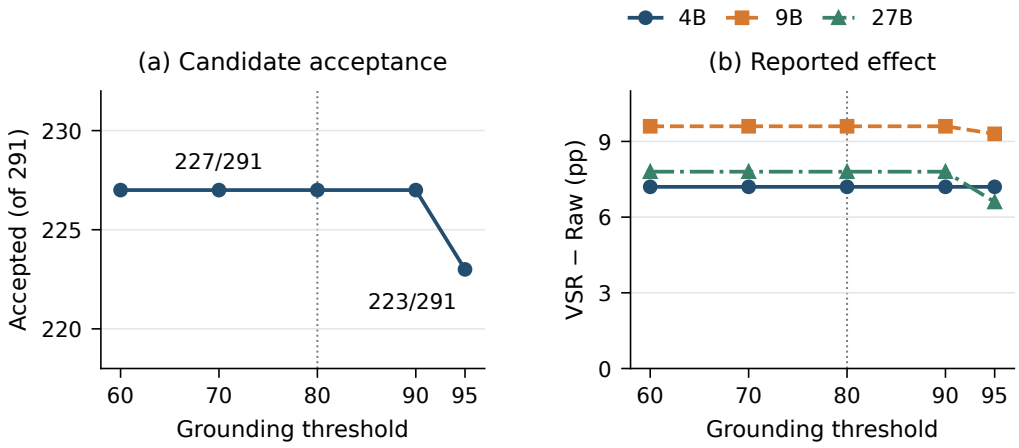


Figure 6: ScienceWorld Grounding sensitivity at five thresholds (60, 70, 80, 90, and 95). Left: accepted candidates among 291 Grounding-applicable test variations. Right: success-rate gains of threshold-routed VSR over Raw Overlay on all 332 held-out test variations. Accepted cases use their VSR-overlay outcomes; all remaining cases use Raw-overlay fallback. Both panels reuse fixed diagnostic and execution results rather than new model runs at each threshold.

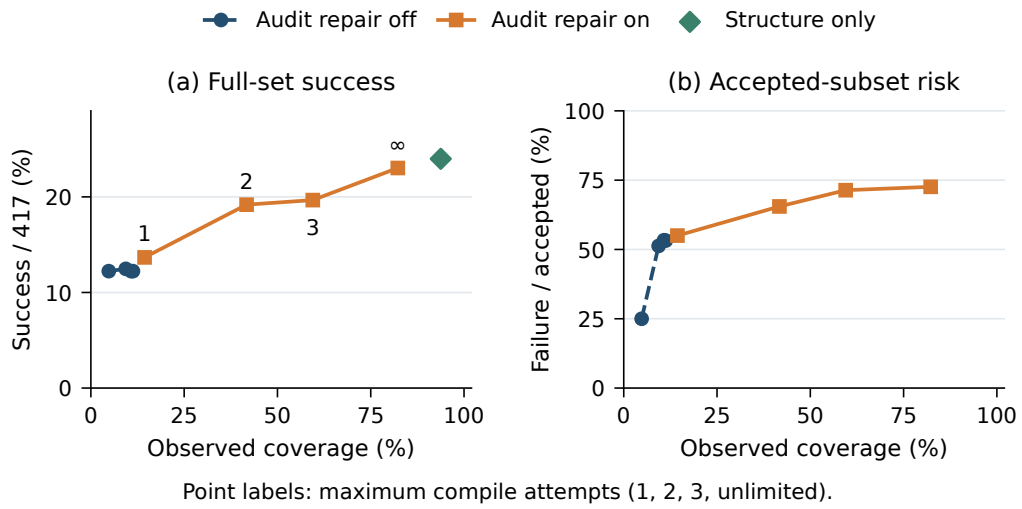


Figure 7: AppWorld verifier coverage and outcome risk. Full-set success retains all 417 tasks; accepted-subset failure uses each configuration's accepted count. The structure-only point appears in the full-set success panel.

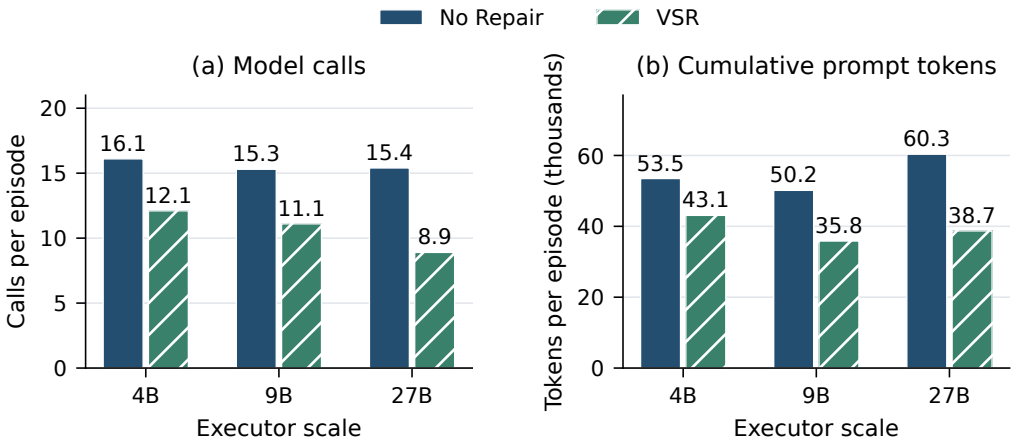


Figure 8: ScienceWorld cost per episode across 332 test variations per executor. VSR reduces calls and cumulative prompt tokens. Token counts cover the whole episode.

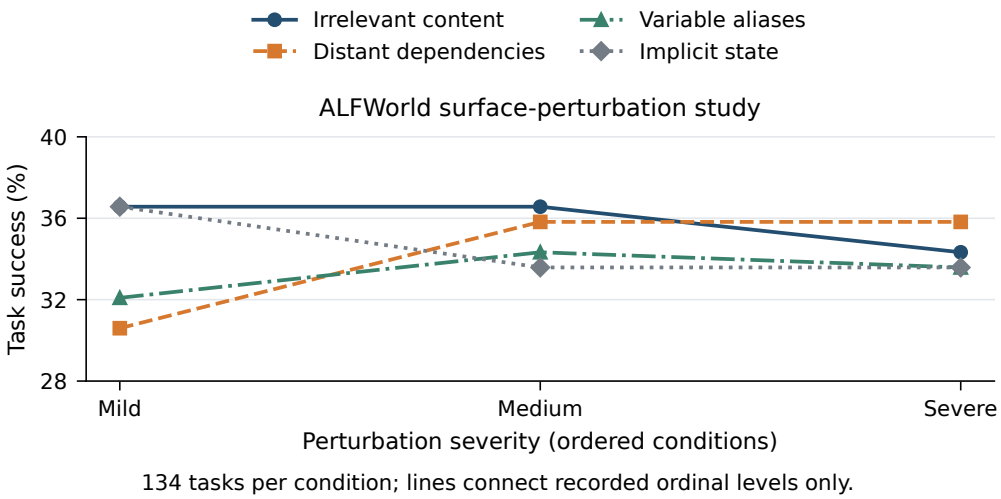


Figure 9: Surface-perturbation dose study. Each point uses 134 tasks; lines connect ordinal severity levels. No monotone dose law is established.

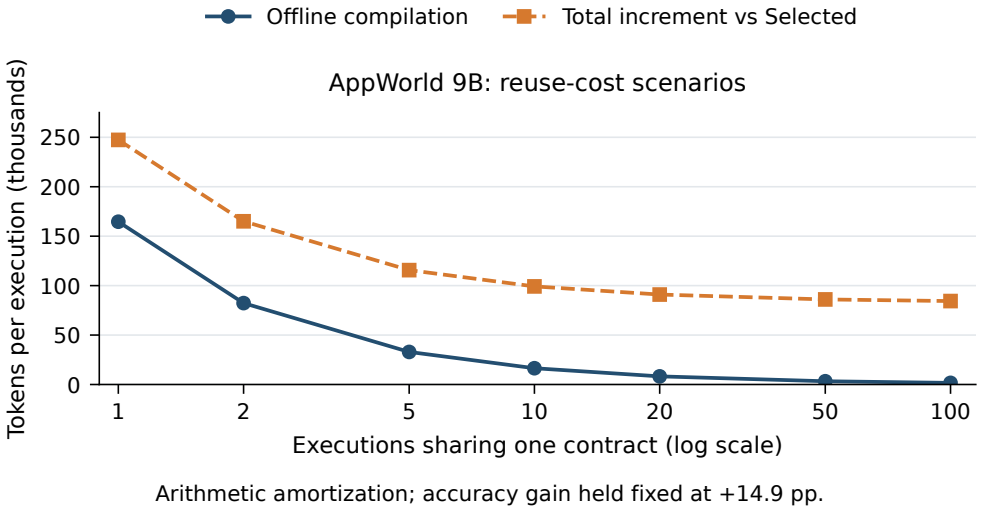


Figure 10: AppWorld cost-amortization scenarios. The logarithmic horizontal axis is the number of executions sharing one compiled contract. Each scenario amortizes compilation cost while holding the measured accuracy gain fixed.

O CROSS-MODEL EVALUATION: GEMMA, GEMINI, AND DEEPSEEK

The additional evaluations test the same two barriers emphasized in the main text: whether the executor can use task-relevant evidence and whether it can turn that evidence into current-instance actions. We evaluate Gemma on ScienceWorld, ALFWorld, and MedCalc, and Gemini 3.8 Flash and DeepSeek V4.1 Flash on ScienceWorld.

O.1 FROZEN PROTOCOL AND EXECUTOR-SPECIFIC DEVELOPMENT

The ScienceWorld library contains demonstrations drawn only from training variations and covers 30 official task families. The held-out evaluation starts after an offset of 20 and contains 332 distinct variations from the 19 families with remaining examples. Each new executor runs these same variations under three conditions, for 996 episodes per executor. The temperature is zero and the maximum episode length is 40 steps. No Repair supplies the original training Skill. Raw Overlay retains the complete library and retrieved trace and uses the executor gate without replacing instance slots. VSR adds only task bindings that pass the provenance, uniqueness, target-grounding, and closure checks. Raw Overlay and VSR therefore share the retrieval material and executor-gate rule.

Gemma uses eight local serving endpoints, with thinking disabled and no external generation API calls. The method and hyperparameters are inherited from the prior protocol, but each Gemma executor has its own 89-task development evaluation (Table 45). Only these development outcomes choose the deployment condition; the test begins after the policy is written to `frozen-policy.json`.

Table 45: Gemma ScienceWorld development outcomes. Entries are successes out of 89. The final column gives the selected deployment policy.

Executor	No Repair	Raw Overlay	VSR	Frozen choice
Gemma-4-12B	19	26	29	`safe_router_overlay`
Gemma-4-31B	21	25	33	`safe_router_overlay`

The API studies reuse the frozen method, training library, held-out tasks, and evaluation protocol. Gemini's IdeaLab route generates hidden reasoning and uses a 256-token output budget. DeepSeek disables thinking explicitly and uses a 16-token output budget. Within an executor, all three conditions have the same budget; cross-provider comparisons involve different service configurations. Parse and execution errors remain in the denominator.

O.2 SCIENCEWORLD PAIRED EFFECTS AND TASK-FAMILY UNCERTAINTY

The main text's ScienceWorld table combines all seven executors. Table 46 gives the additional paired effects and family-bootstrap intervals. McNemar tests use episode-level paired outcomes. The bootstrap intervals cluster by task family. All p values are unadjusted. Gemma success counts uniquely reconstructed from the success rates and $N = 332$ are 106/121/147 for 12B and 100/112/142 for 31B (No Repair/Raw/VSR). Direct episode counts are 172/171/187 for Gemini and 105/109/135 for DeepSeek.

The direction of VSR–Raw improvement is shared across the seven executors, but its task-family robustness is heterogeneous. Gemma-31B's interval remains positive; Gemma-12B's and Gemini's cross zero, while DeepSeek's lower endpoint touches zero. Gemini's comparison with No Repair also falls short of episode-level $p < 0.05$. Its mean native environment score is 1.20 points below Raw despite its higher success rate (Table 47).

O.3 GROUNDING DIAGNOSIS, ACCEPTANCE, AND ABSTENTION

For each API executor, 291 variations have applicable Grounding candidates out of 332, leaving 41 outside the applicable candidate set. Of those 291 candidates, 227 pass verification and 64 cause abstention. The mean Grounding certificate score changes from 47.4 to 91.9 for Raw versus candidate representations; these are specialized slot-certificate scores, not the overall scalar SCS. Table 48 relates accepted candidates to observed execution outcomes.

Table 46: Additional ScienceWorld paired effects. W/L denotes successes unique to VSR/control; — denotes unavailable pair counts. Confidence intervals are family-bootstrap intervals in percentage points.

Executor	Control	Gain (pp)	W/L	Paired p	Family 95% CI (pp)
Gemma-4-12B	No Repair	+12.3	63/22	9.8×10^{-6}	$[+0.6, +22.1]$
	Raw Overlay	+7.8	—	0.0019	$[-1.0, +17.9]$
Gemma-4-31B	No Repair	+12.7	53/11	1.0×10^{-7}	$[+3.7, +21.3]$
	Raw Overlay	+9.0	—	5.3×10^{-6}	$[+2.9, +16.9]$
Gemini 3.8 Flash	No Repair	+4.5	35/20	0.0581	$[-4.0, +13.2]$
	Raw Overlay	+4.8	35/19	0.0402	$[-2.1, +12.4]$
DeepSeek V4.1 Flash	No Repair	+9.0	53/23	0.000765	$[+1.1, +15.2]$
	Raw Overlay	+7.8	49/23	0.00294	$[0.0, +16.6]$

Table 47: Additional ScienceWorld native scores and episode cost. Gemma prompt reductions compare cumulative episode tokens with No Repair. Dashes denote unreported values. API score entries are differences, not absolute scores.

	Mean native score			Mean steps		Prompt
Executor	No	Raw	VSR	No	VSR	reduction
Gemma-4-12B	7.2	13.3	18.7	18.7	15.3	30.7%
Gemma-4-31B	23.1	27.9	30.9	23.0	19.7	12.1%

Executor	Mean score: VSR–No Repair	Mean score: VSR–Raw
Gemini 3.8 Flash	+0.98	−1.20
DeepSeek V4.1 Flash	+7.72	+0.65

Table 48: API-model Grounding diagnosis and execution on the 227 accepted cases. Both models have 291 applicable candidates, 227 accepted candidates, and 64 abstentions out of 332 total variations.

Executor	Raw/candidate score	VSR–Raw (pp)	W/L	Paired p
Gemini 3.8 Flash	47.4/91.9	+6.2	26/12	0.0336
DeepSeek V4.1 Flash	47.4/91.9	+11.0	46/21	0.00307

Abstention retains source evidence and does not remove a task from evaluation. Separately run fallback episodes can have different outcomes: subtracting the accepted-subset pairs from the full-test pairs gives 9/7 VSR/Raw wins on the remaining 105 Gemini cases and 3/2 on the remaining DeepSeek cases. These counts are derived from the aggregate pairs.

O.4 All held-out task families for the API models

Tables 49 and 50 include every held-out family. Gemini gains 16 successes over Raw overall, of which 13 come from `test-conductivity`; that family also accounts for all 15 of its net successes over No Repair. DeepSeek gains 26 over Raw, including 15 in `test-conductivity`. These distributions explain why aggregate paired improvements coexist with wide task-family intervals.

Table 49: Gemini 3.8 Flash: successes in all 19 ScienceWorld held-out families. Differences are success counts, not percentage points.

Task family	N	No	Raw	VSR	ΔNo	ΔRaw
find-animal	20	19	20	20	+1	0
find-living-thing	20	19	20	20	+1	0
find-non-living-thing	20	20	20	20	0	0
find-plant	20	17	13	15	−2	+2
grow-fruit	13	0	0	0	0	0
grow-plant	13	4	0	0	−4	0
inclined-plane-determine-angle	20	0	0	2	+2	+2
inclined-plane-friction-named-surfaces	20	7	6	9	+2	+3
inclined-plane-friction-unnamed-surfaces	20	3	5	5	+2	0
lifespan-longest-lived	12	12	12	12	0	0
lifespan-longest-lived-then-shortest-lived	12	11	11	11	0	0
lifespan-shortest-lived	12	11	11	11	0	0
measure-melting-point-known-substance	20	2	2	1	−1	−1
measure-melting-point-unknown-substance	20	8	6	6	−2	0
mendelian-genetics-known-plant	10	2	3	3	+1	0
mendelian-genetics-unknown-plant	20	5	4	7	+2	+3
test-conductivity	20	5	7	20	+15	+13
test-conductivity-of-unknown-substances	20	8	13	8	0	−5
use-thermometer	20	19	18	17	−2	−1
Total	332	172	171	187	+15	+16

O.5 Gemma ALFWorld: transferring an execution scaffold

Both Gemma executors run the same 134 `valid_unseen` tasks. VSR combines an observation-driven execution frontier with repeated-action guards and model retries (Table 51). Contracts are compiled offline by local Qwen3.5-27B from visible task text and six generic operation schemas, without hidden environment trajectories. These results test transfer of compiled VSR artifacts to new executors using the combined execution scaffold.

O.6 Gemma MedCalc: accuracy and evidence-selection cost

Each executor receives 800 paired held-out inputs under Full Skill and evidence-selection VSR (Table 52). Neither paired accuracy difference is statistically detected. The prompt and latency reductions are point estimates.

O.7 Cross-model execution cases

In the paired Gemma ScienceWorld episode `find-animal__v258`, the changed binding is `kitchen` → `bedroom`; the slot certificate changes from 50 to 100. No Repair starts with `look around` and fails at score −100. Raw Overlay starts with `teleport to kitchen` and fails at score 83; VSR starts with `teleport to bedroom` and succeeds at score 100.

Table 50: DeepSeek V4.1 Flash: successes in all 19 ScienceWorld held-out families. Differences are success counts, not percentage points.

Task family	N	No	Raw	VSR	ΔNo	ΔRaw
find-animal	20	14	20	20	+6	0
find-living-thing	20	11	20	18	+7	−2
find-non-living-thing	20	4	8	10	+6	+2
find-plant	20	6	7	7	+1	0
grow-fruit	13	0	0	0	0	0
grow-plant	13	1	0	0	−1	0
inclined-plane-determine-angle	20	9	8	11	+2	+3
inclined-plane-friction-named-surfaces	20	9	4	10	+1	+6
inclined-plane-friction-unnamed-surfaces	20	11	9	13	+2	+4
lifespan-longest-lived	12	9	9	10	+1	+1
lifespan-longest-lived-then-shortest-lived	12	0	0	0	0	0
lifespan-shortest-lived	12	3	3	4	+1	+1
measure-melting-point-known-substance	20	0	0	0	0	0
measure-melting-point-unknown-substance	20	0	0	0	0	0
mendelian-genetics-known-plant	10	2	1	1	−1	0
mendelian-genetics-unknown-plant	20	7	7	7	0	0
test-conductivity	20	7	1	16	+9	+15
test-conductivity-of-unknown-substances	20	11	11	8	−3	−3
use-thermometer	20	1	1	0	−1	−1
Total	332	105	109	135	+30	+26

Table 51: Gemma ALFWorld results, 134 paired tasks per executor. Success counts are uniquely reconstructed from the reported percentages and denominator. Retries are total additional guard-triggered model requests.

Executor	Static Skill	VSR State	Gain (pp)	W/L	Paired p
Gemma-4-12B	38/134 (28.4%)	117/134 (87.3%)	+59.0	80/1	6.8×10^{-23}
Gemma-4-31B	115/134 (85.8%)	124/134 (92.5%)	+6.7	12/3	0.035

Executor	Static mean steps	VSR mean steps	Guard retries
Gemma-4-12B	39.1	21.5	536
Gemma-4-31B	18.7	16.8	213

Table 52: Gemma MedCalc results. Counts are uniquely reconstructed from the reported rates and $N = 800$. Efficiency entries are reported reductions relative to Full Skill.

Executor	Full Skill	Selection VSR	Gain (pp)	W/L	Paired p
Gemma-4-12B	588/800 (73.5%)	585/800 (73.1%)	−0.4	53/56	0.85
Gemma-4-31B	640/800 (80.0%)	640/800 (80.0%)	0.0	36/36	1.00

Executor	Prompt reduction	Latency reduction
Gemma-4-12B	11.8%	34.0%
Gemma-4-31B	11.8%	31.5%

The Gemma ALFWorld case has identifier

`pick_clean_then_place_in_recep-Cloth-None-CounterTop-424`.

The static run takes a towel instead of a cloth and fails at the 50-step limit. Its prefix is `go to towelholder 1`, `take towel 1 from towelholder 1`, `go to sinkbasin 1`, `examine sinkbasin 1`, `look`, and `examine towel 1`. The successful VSR trace has ten actions: `go to towelholder 1`; `go to handtowelholder 1`; `go to handtowelholder 2`; `go to cabinet 1`; `open cabinet 1`; `take cloth 1 from cabinet 1`; `go to sinkbasin 1`; `clean cloth 1 with sinkbasin 1`; `go to countertop 1`; and `move cloth 1 to countertop 1`. Observations rule out searched locations before cleaning and placement proceed.

The API-model cases illustrate executions in which VSR succeeds and Raw fails. In Gemini's `find-plant__v253`, the verified substitutions are `blue` → `green` and `workshop` → `living room`, each occurring once. The Grounding certificate changes from 50 to 100. No Repair and VSR succeed at score 100; Raw fails at 75. The Raw prefix is `go to outside`, `go to greenhouse`, `focus on adult pea plant`, `pick up flower pot 9`, `go to hallway`, and `go to workshop`. The VSR prefix is identical except that its sixth action is `go to living room`; the No Repair prefix instead moves the plant to inventory before going to the hallway.

In DeepSeek's `find-non-living-thing__v247`, the single verified substitution is `living room` → `bathroom`, and the Grounding certificate again changes from 50 to 100. No Repair fails at score −100 after the action `focus on egg giant tortoise`; Raw fails at 75 after going through the kitchen and hallway to the living room, looking around, focusing on a book, and emitting the literal action `0`. VSR succeeds at 100 with the prefix `go to kitchen`, `go to hallway`, `teleport to bathroom`, `look around`, `focus on finger painting`, and `move finger painting to green box`.

O.8 Costs and execution failures

Table 53: API usage totals and estimated charges. DeepSeek totals include smoke/development prechecks. Completion and reasoning token counts are listed separately and may overlap.

Quantity	Gemini 3.8 Flash	DeepSeek V4.1 Flash
API requests	15,572	12,949
Prompt tokens	75,488,052	47,129,974
Completion tokens	2,523,500	12,949
Reasoning tokens	2,498,239	0
Cache-hit rate	0.0%	64.5%
Estimated charge (USD)	66.08	2.61
Estimated charge (CNY)	470	19
Peak-price, all-cache-miss upper estimate (USD)	—	14.15
Peak-price, all-cache-miss upper estimate (CNY)	—	101

The Gemini estimate uses public pricing; the IdeaLab bill may differ. DeepSeek's estimate accounts for peak/off-peak pricing and observed cache hits. These usage-based estimates reflect different reasoning configurations and output budgets.

Table 54: ScienceWorld completeness and execution checks for the added models. Each row has 332 result files in each condition. Error counts are No Repair/Raw/VSR and remain in the success-rate denominator.

Executor	Errors (No/Raw/VSR)	Gate actions	Gate mismatches
Gemma-4-12B	0/0/6	3,121	0
Gemma-4-31B	0/1/0	3,328	0
Gemini 3.8 Flash	3/1/1	3,129	0
DeepSeek V4.1 Flash	0/0/0	3,065	0

The Gemma ScienceWorld errors are invalid action indices: six for 12B VSR and one for 31B Raw. MedCalc has one or two parsing failures, without a model-by-condition breakdown, and one truncation per 12B condition. Both Gemma runs record zero thinking characters and zero external generation calls.