CPUNESY: CONTROLLING MODEL WRITES FOR RELIABLE NEURO-SYMBOLIC REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models are highly effective at recalling statistical patterns, but their accuracy degrades sharply when answers must be derived rather than recalled, especially on deep multi-hop chains. Delegating derivation to deterministic symbolic executors shifts the reliability question to whether model-generated premises are supported by the source. We introduce CPUNeSy (Certified Predicate-Unit Neuro-Symbolic Networks), a serving architecture that controls model writes to symbolic state through a task-defined predicate interface and a certificate gate, and abstains when grounding passes disagree. A component analysis separates deterministic execution from restricted grounding, agreement, and source rechecking. Our experiments show that deterministic execution accounts for most of the accuracy recovery on derivation-heavy tasks, whereas controlled writes primarily improve selective reliability by withholding unsupported or inconsistent answers at the cost of coverage. On controlled multi-hop stress tests in law and formal mathematics, deterministic execution recovers most of the accuracy gap over chain-of-thought and retrieval baselines, with full-pool gains up to 35.0 points. Certification behaves as a selective-serving control rather than an accuracy mechanism: with grounding traces held fixed on ContractNLI, source rechecking removes a quarter of DeepSeek's wrong answers that survive two-vote agreement, at a measurable coverage cost. When abstention is expensive, routing withheld cases to an explicitly uncertified same-model fallback raises full-pool accuracy on MedCalc-Bench Verified by 13.9 and 4.9 points for Seed and DeepSeek; these gains do not belong to the certified serving channel. On LeanDojo Benchmark 4, kernel-restricted pools match BM25 recall@15 (89.3%). Gains depend on the grounding model's error regime: bias-dominated grounders benefit less, consistent with the voting bound we state formally. The resulting certificates guarantee derivational validity relative to admitted premises; semantic faithfulness to natural-language sources remains conditional on the source checker, and prospective validation of the deployment protocol is future work.

1 INTRODUCTION

Large language models recall facts remarkably well (Kaplan et al., 2020; Srivastava et al., 2023), yet on tasks whose correctness depends on *composing* several rules or facts from a fixed source of truth, token generation often imitates a derivation rather than executing one (Dziri et al., 2023; Kordjamshidi et al., 2026). On real mathematical dependency graphs and multi-hop legal reasoning, chain-of-thought and retrieval baselines plateau well below what the underlying formal structure allows, and increasing the reasoning budget or the number of sampled traces yields diminishing returns (Wei et al., 2022; Lewis et al., 2020; Wang et al., 2023). The central difficulty is ensuring that an answer follows from the stated premises: a chain-of-thought trace can appear coherent even when it skips a necessary inference or introduces an unsupported intermediate conclusion. Legal, medical and formal-mathematics deployments make this concrete, since a downstream decision cannot rest on an answer whose intermediate steps are neither auditable nor tied to source.

A natural response is to delegate derivation to a symbolic executor once the model has proposed the relevant facts and rules (Pan et al., 2023; Gao et al., 2023). Execution is deterministic and cheap to check, and it turns a multi-hop question into a query against a well-defined logical state. But execution alone is not a correctness certificate: a valid derivation from a premise that misrepresents

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

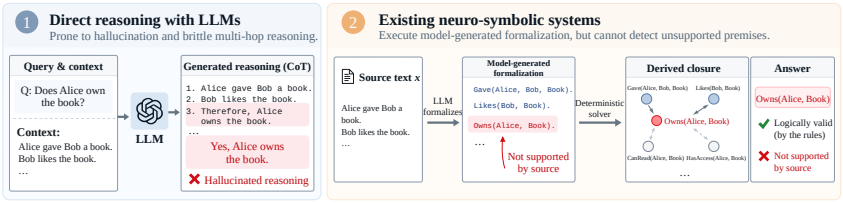


Figure 1: Two routes to the same wrong answer: hallucinated reasoning (*left*) and valid execution of unsupported premises (*right*). These failure modes motivate controlling what language models may write into symbolic state.

the source produces a wrong answer just as convincingly as a hallucinated chain (Figure 1). A model can arrive at the same wrong answer either by fabricating an intermediate conclusion in a chain of thought or by writing it into the solver's premise set and letting the executor rediscover it. To rule this out, the system needs to decide which model-proposed premises may enter its state at all, and to keep evidence for every accepted premise so that a served answer can be traced end-to-end.

Neuro-symbolic architectures combine learned and symbolic components (Besold et al., 2022; Colelough & Regli, 2025), and recent LLM-plus-solver pipelines (Olausson et al., 2023; Ye et al., 2023; Jiang et al., 2024; Sun et al., 2024; Xu et al., 2024; Li et al., 2024a; 2025; 2026) inherit the correctness of their solver but expose it to whatever the parser writes: admission is implicit, and an ill-supported premise is checked only by whether the solver can still run. CPUNeSy (Certified Predicate-Unit Neuro-Symbolic Networks) treats admission as a first-class step (Figure 2). Under the architecture's contract, the model proposes predicate units, each a fact or a Horn rule; a certificate gate checks their source support and rejects proposals whose target predicate lies outside the allowed source vocabulary; a deterministic executor derives the answer; and every certified answer carries a bounded-depth derivation replayable against the current state. Restricting the source vocabulary prevents the model from inserting an intermediate conclusion, or the target answer, as a source fact, so the executor is the only route by which a rule-head predicate can enter state.

Studying such a system requires disentangling several mechanisms that change at once when execution is added on top of a language model: derivation shifts from sampling to solving, the model is queried in a more structured way, and repeated calls are aggregated by voting. Our experiments separate these. We compare a language model alone, the same model paired with a deterministic solver, and the restricted two-vote policy, separating the contribution of execution from the additional serving policy. We then hold both grounding traces fixed on the same benchmark and compare plain two-vote agreement against agreement plus source rechecking, so the difference quantifies what source checking adds over voting itself. Finally, an offline admission replay tests the vocabulary barrier with injected target predicates; the historical restricted runs enforced that vocabulary in the prompt. Five suites span legal reasoning, formal mathematics, and medical calculation; controlled multi-hop tests measure execution under adversarial mid-link-break negatives, public ContractNLI and MedCalc comparisons expose how admission trades errors against abstentions, and a matched LeanDojo comparison tests kernel-based premise pools against BM25 and a learned retriever. We report answered accuracy, coverage, and full-pool accuracy that counts every abstention as an error. Two findings organize the analysis. First, deterministic execution accounts for most of the accuracy recovery on derivation-heavy tasks, so a one-vote solver already closes most of the gap. Second, controlled writes primarily improve selective reliability, withholding unsupported or inconsistent answers at the cost of coverage; their value is the error–abstention tradeoff they impose on top of execution, not a further accuracy gain. The resulting certificates are for derivational validity relative to admitted premises, and semantic faithfulness to natural-language sources remains conditional on the source checker. Our contributions are as follows:

- A checked interface that separates model proposals from symbolic state updates, with formal guarantees conditional on source-checker and verifier soundness.
- A component analysis separating deterministic execution from restricted grounding, agreement, and source rechecking, including a fixed-trace gate ablation on public ContractNLI data.
- An empirical characterization of the resulting error–coverage–cost tradeoffs, with diagnostics for when controlled writes and optional kernelization are useful.

2 Related Work

LLM-to-symbolic solver pipelines. The natural way to combine a language model with a symbolic executor is to let the model translate and the solver execute. Logic-LM (Pan et al., 2023), LINC (Olausson et al., 2023), and SatLM (Ye et al., 2023) established this pattern for first-order logic, SAT, and constraint solving, and LeanReasoner (Jiang et al., 2024) and CLOVER (Ryu et al., 2025) carry it into proof assistants and verification-backed domains. Once translation worked often enough to be useful, the bottleneck moved to its failures. One-shot formalization is brittle, so DetermLR (Sun et al., 2024), LINA (Li et al., 2024a), and HBLR (Li et al., 2025) wrap it in validate-and-repair loops, and SymbCoT (Xu et al., 2024), Aristotle (Xu et al., 2025), and CaRing (Yang et al., 2025) keep the reasoning trajectory itself in symbolic form so that errors can be caught mid-chain. Across these refinements, one thing did not change. Once a formal problem is written down, execution is trusted, and whether the written premises are supported by the source is never asked. CPUNeSy asks exactly that question, checking source support before a proposed predicate may become a premise for execution.

Neuro-symbolic engines, provenance, and structured generation. A second body of work brings symbolic structure into learning rather than serving. DeepProbLog (Manhaeve et al., 2018), Logic Tensor Networks (Badreddine et al., 2022), NeurASP (Yang et al., 2020), Scallop (Li et al., 2023), LNN (Riegel et al., 2020), and SATNet (Wang et al., 2019) each make a different fragment of logic differentiable or probabilistic, so that gradients can flow through constraints during training. Alongside them, constrained decoding made formal output syntactically reliable, first for SQL with PICARD (Scholak et al., 2021) and then as general infrastructure with XGrammar (Dong et al., 2025), and Datalog provenance (Bourgaux et al., 2022) tracks which base facts support each derived fact.

Neural theorem proving. Theorem proving followed a similar arc. GPT-f (Polu & Sutskever, 2020) showed that a language model can predict useful proof steps, retrieval made premise selection practical at library scale through Magnushammer (Mikuła et al., 2024) and LeanDojo's ReProver (Yang et al., 2023), and COPRA (Thakur et al., 2024) turned proving into an agentic search over tactic applications. Recent work also verifies intermediate natural-language claims (Liu et al., 2025). Our premise pools borrow the premise-retrieval setting but change the object being retrieved, since the pool is built from the source kernel and its dependency closure rather than a raw library. Section 4.1 compares them with ReProver and BM25 directly.

Selective prediction, verification, and benchmarking. Knowing when not to answer has its own history. Selective prediction lets a model abstain below a confidence threshold (Geifman & El-Yaniv, 2017), and post-hoc verification scores or repairs answers after generation (Cobbe et al., 2021; Liu et al., 2025). A growing set of benchmarks, LogicBench (Parmar et al., 2024), ProverGen/ProverQA (Qi et al., 2025), RuleArena (Zhou et al., 2025), SLR (Helff et al., 2026), LogicGraph (Wu et al., 2026), MetaPCR (Galitsky et al., 2026), and Reasoners (Kordjamshidi et al., 2026), exists because the community kept discovering that fluent systems reason incorrectly. CPUNeSy shares the abstention instinct but moves the check earlier, before execution rather than after generation, and the fixed-trace comparison in §4.2 separates what source checking adds from what voting already provides. Appendix A gives the full comparison.

3 The CPUNeSy Method

3.1 Architecture

A *predicate unit* is either a ground fact or a Horn rule. CPUNeSy uses these units to represent both source premises and derived conclusions. A model proposes units and inference steps, and a checker decides which proposals may enter the state. This addresses two common failure modes. A model that produces an answer and rationale together may skip a required inference. A model that writes a formal problem for a solver may introduce facts or rules unsupported by the source. Figure 1 illustrates the two failure modes, while Figure 2 shows how CPUNeSy separates model proposals from checked state updates.

A CPUNeSy instance has three components. The language model $\mathcal{N}$ maps source text to facts and rules, a step called *grounding*, and verbalizes the results. The checking interface $\mathcal{B}$ extracts and

verifies inference steps. The symbolic core stores a *kernel*, a set of source units with no redundant members, and a cache of verified conclusions. We refer to the model and checking interface as the carrier and bridge in the formal definitions. The architecture specifies two phases (Figure 2). During DEPLOY, candidate facts and rules are checked for type and source support. A deletion scan in a fixed order removes each unit that can be derived from the remaining units. During SERVE, a query is grounded against that kernel, candidate derivations are checked edge by edge, and only a certified result is verbalized. For example, if the checked source contains p and $p \to q$, the system may derive and serve q with both units in its certificate. The source kernel remains $\{p, p \to q\}$, from which q can be reconstructed.

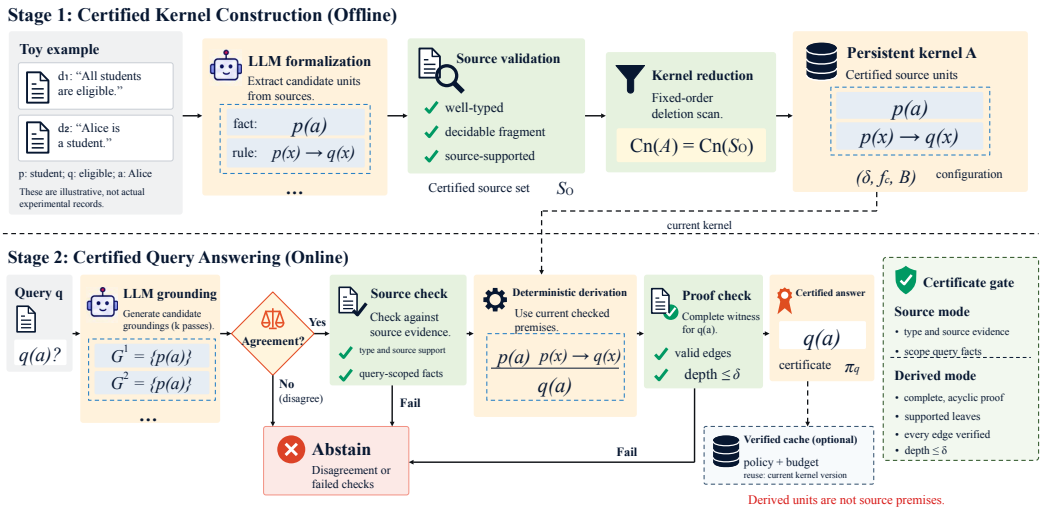


Figure 2: The CPUNeSy architecture contract. *Stage 1 (offline):* DEPLOY formalizes and validates source units, then reduces the certified source set S_O to an irredundant kernel A with $\mathrm{Cn}(A) = \mathrm{Cn}(S_O)$. *Stage 2 (online):* SERVE checks agreeing query groundings, derives an answer, and verifies its proof before release.

The certificate gate uses separate checks for source premises and derived conclusions. In SOURCE mode the gate checks a proposed fact or rule against its source record and adds supported units to a certified source set; a canonical deletion scan then recomputes the kernel and omits units that the remainder entails, which matters because a later source unit can make an earlier one redundant. In DERIVED mode the gate checks that every edge of a proposed derivation is valid over the current kernel and that its certified depth is within budget; a verified conclusion may then be served or cached while retaining its dependencies on the source premises. Repeated grounding filters proposals on which the model's calls disagree, and Algorithm 1 states both modes together. Table 1 then specifies the checks used in each domain: the mathematical experiments execute recorded proof-library dependency graphs, while the natural-language tasks rely on schema- and vocabulary-grounded checks whose extraction step is model-mediated. The offline admission replay enforces the leaf-predicate restriction and preserves all recorded answers from that interface (Appendix G).

The experiments instantiate different parts of this contract. Historical L1-HARD runs restricted the vocabulary in the prompt; an explicit admission barrier is evaluated separately by offline replay. Complete per-query certificate archival was not instrumented in every adapter. Table 7 and Appendix G distinguish these implementation scopes.

At time t, the core contains source facts A_t, source rules $\mathcal{R}_t$, cached consequences, and a certificate log. It starts empty or from a source-checked kernel with an empty cache. Every addition is checked. Erasure and eviction remove cached results that depend on a removed unit, so the state can both grow and shrink. After a kernel change, each retained cache certificate must be rechecked or rebuilt over the new kernel within depth δ; entries that fail this check are evicted. Certificates record complete derivation DAGs and transitive source dependencies. Cache reuse is allowed only against the current kernel version. The persistent source store holds the kernel. Derived conclusions are recomputed or cached with their proof dependencies. The kernel result in Theorem 5 shows when this omission

Table 1: Source checks used in experimental adapter and extraction errors they can leave unresolved.

Adapter	Source checks	Possible extraction errors
L1-HARD	Rule-derived leaf vocabulary; parsed facts; agreement of solver answers	LLM maps narrative to leaf facts; historical vocabulary restriction was in prompt
ContractNLI	Schema-grounded facts; LLM source recheck of contributing facts; two verdicts	Rechecker can share the grounder's semantic bias
MedCalc	Calculator schema, types and units; equality of normalized attribute maps; executable calculation	Schema-valid values can still be incorrectly extracted
M-HARD / LD-MH	Candidate premise IDs; dependency closure in a frozen proof-library DAG	Mapping to IDs and completeness of the recorded graph

Algorithm 1 Certificate gate with separate source and derived modes

Require: mode $m \in \{\textsc{Source}, \textsc{Derived}\}$; proposed unit u with evidence ξ; certified source set S_t; core $\mathcal{C}_t = (A_t, \mathcal{R}_t, \mathrm{Cache}_t)$; depth budget δ; source checker V_{src}; edge extractor E; fragment verifier V
Ensure: `SOURCE-ADMIT`, `REDUNDANT`, `CERTIFY`, or `REJECT`, with a checkable record

$\Gamma_t \leftarrow A_t \cup \mathcal{R}_t$
if $m = \textsc{Source}$ **then**
  if u is ill-typed **or** $V_{\mathrm{src}}(u, \xi) = \bot$ **then**
    return `REJECT` (invalid or unsupported source unit)
  end if
  $\pi_{\mathrm{src}}(u) \leftarrow \langle u, \xi, V_{\mathrm{src}}\text{-log}\rangle$
  $S_{t+1} \leftarrow S_t \cup \{u\}$; $(K_{t+1}, J_{t+1}) \leftarrow \textsc{KernelScan}(S_{t+1})$
  partition K_{t+1} into facts A_{t+1} and rules $\mathcal{R}_{t+1}$
  log $\pi_{\mathrm{src}}(u)$; recheck or rebuild cache certificates over K_{t+1} within δ; evict failures
  if $u \in J_{t+1}$ **then**
    return `REDUNDANT` with $\pi_{\mathrm{src}}(u)$ ▷ certified, omitted from core
  end if
  return `SOURCE-ADMIT` with $\pi_{\mathrm{src}}(u)$
end if
$\mathcal{E}(u) \leftarrow E(u, \xi)$ ▷ ξ is the carrier trace
if $\mathcal{E}(u)$ is not a complete acyclic derivation of u over Γ_t **then**
  return `REJECT` (unlinked conclusion or unsupported leaf)
end if
for all $e \in \mathcal{E}(u)$ in topological order **do**
  if $V(e, \Gamma_t) = \bot$ **then**
    return `REJECT` (unverifiable derivation edge e)
  end if
end for
$d \leftarrow \mathrm{depth}(\mathcal{E}(u))$ ▷ depth of the verified witness
if $d > \delta$ **then**
  return `REJECT` (depth budget exceeded)
end if
$\kappa \leftarrow \mathrm{key}(u)$; $\pi_{\mathrm{der}}(u) \leftarrow \langle \mathcal{E}(u), d, \kappa, \mathrm{deps}(u), V\text{-log}\rangle$
return `CERTIFY` with $\pi_{\mathrm{der}}(u)$ ▷ serve; cache only by policy

preserves *closure*, the set of conclusions derivable from the source. The store–recompute threshold in Proposition 3 identifies units eligible for caching; the scheduler then selects a subset that fits the storage budget. Section 3.2 states the interface properties of this design formally (Theorem 1) and the assumptions under which they extend to coverage, storage, and reliability guarantees.

3.2 Formal Guarantees and Operating Conditions

Let $\mathbb{S}_O$ denote the certified source set, A its fixed-order irredundant kernel, and Cn closure in a decidable Horn fragment. The guarantee requires both valid inference and correct source interpretation. Theorem 1 states the soundness assumptions on the two checkers. Table 1 describes how the experimental source checkers implement these checks.

Theorem 1 (Certified-state soundness)**.** *Assume that the source checker is sound for the declared grounding interface and that the fragment verifier is sound. Starting from the initialization in*

Section 3.1, after any sequence of its transitions, (i) every persistent premise has a recheckable source-support record; (ii) every consequence u served at time t or cached in that state satisfies $u \in \mathrm{Cn}(A_t \cup \mathcal{R}_t)$ and $D_{\mathrm{d}}^{\mathrm{cert}}(u \mid A_t \cup \mathcal{R}_t) \leq \delta$; and (iii) erasing a source unit removes every cached consequence whose certificate depends on it.

The induction proof is in Appendix D.2. Each admitted premise retains its evidence and checker version, and each derived unit retains a complete witness over the current kernel. The source-soundness assumption is not established for the experimental natural-language checkers, whose residual risk is measured empirically. The mathematical adapters execute recorded dependency graphs; they do not constitute an evaluation of end-to-end proof search.

Proposition 1 (No promotion through restricted writes)**.** *Fix a rule set $\mathcal{R}$ and a source vocabulary Ω disjoint from the predicates in rule heads, with all initial source facts in Ω. Let the admission barrier reject any proposed source fact outside Ω. Then adding an intermediate or target predicate to a proposal cannot admit that predicate as a source premise. Every accepted derivation of such a predicate must contain a rule application. This property holds for arbitrary model proposals and does not assume statistical independence or semantic accuracy of allowed leaf predicates.*

Proof. A rule-head predicate is outside Ω, hence fails admission. It can enter closure only through the consequence of a rule. Rejected proposals contribute no premises. The same argument applies after every query-scoped reset. □

Restricting source writes blocks direct insertion of the desired conclusion as a premise. Allowed leaf facts still require source checking. Appendix G examines these facts and tests the restriction by adding target conclusions to recorded proposals.

Lemma 1 (Voting under shared bias and independent slips)**.** *Let β_{ans} be the query mass of a shared-error regime. Conditional on each query in this regime, suppose traces independently certify the same wrong answer with probability at least $1 - \sigma_c$, where $\sigma_c < 1/2$. On its complement, suppose they independently certify the unique correct answer with probability at least $1/2 + \delta_0$, $\delta_0 > 0$. For k traces under strict-majority serving (otherwise abstaining), the probability $\mathrm{err}(k)$ of serving a wrong answer obeys*

$$\beta_{\mathrm{ans}}\Big(1 - e^{-2k(1/2-\sigma_c)^2}\Big) \;\leq\; \mathrm{err}(k) \;\leq\; \beta_{\mathrm{ans}} + e^{-2k\delta_0^2}.$$

Appendix D.6 gives the conditional bound. With two valid binary answers, majority voting with tie abstention is exactly the deployed agreement rule, and the fixed-trace source-recheck ablation in Section 4.2 measures what checking adds beyond it. For $k = 2$ the bound can be loose; two calls alone neither identify its latent parameters nor establish conditional independence.

The deletion scan preserves $\mathrm{Cn}(A) = \mathrm{Cn}(\mathbb{S}_O)$ (Theorem 5), so the recomputed kernel decides every query that the certified source can decide, and any storage savings arise as a byproduct of whatever redundancy the scan finds in a given corpus.

4 Experiments

Setup. Five suites span three domains. Legal reasoning uses L1-HARD (240 two- to four-hop queries with mid-link-break negatives) and ContractNLI; formal mathematics uses M-HARD (200 depth-2–6 AFP-DAG tasks) and LD-MH; medical calculation uses MedCalc-Bench Verified (1,100 items over 55 verified calculators). Neural controls use chain-of-thought, complete-chain RAG, and high-reasoning-effort reruns at 2.2–3.1× budget. Gradient-trained systems (Manhaeve et al., 2018; Badreddine et al., 2022) require labels absent from this serving-time setting, so the learned-component control is pretrained ReProver on LeanDojo Benchmark 4. **Grounding models.** CPUNeSy uses K3 on controlled stress tests and the model named in each column on the public benchmarks.

4.1 RQ1: Does Symbolic Derivation Recover Accuracy When Answers Must Be Derived?

The public benchmarks separate two ways of serving (Table 2). The native certified channel already leads every baseline on LD-MH and on ContractNLI with Seed, and with the same-model fallback

Table 2: Full-pool accuracy on public benchmarks (%; abstentions count as errors).

Method	ContractNLI		MedCalc-Bench Verified		LeanDojo LD-MH	
	Seed	DeepSeek	Seed	DeepSeek	Seed	DeepSeek
Pure CoT (Wei et al., 2022)	79.6	79.9	66.1	71.5	50.0	51.5
RAG / open-book (Lewis et al., 2020)	73.2	74.7	76.2	85.5	51.0	50.0
CPUNeSy native	81.4	70.9	72.5	63.5	**92.5**	79.0
answered accuracy	87.6	88.7	92.9	95.2	93.9	88.3
native coverage	93.0	79.9	78.0	66.6	98.5	89.5
CPUNeSy + fallback	**84.9**	**83.3**	**90.1**	**90.4**	**92.5**	**83.5**
Logic-LM (Pan et al., 2023)	0.0	0.0	75.5	77.5	50.5	46.5
DetermLR (Sun et al., 2024)	48.4	70.0	67.5	61.5	50.5	49.5
LINC (Olausson et al., 2023)	16.0	0.2	66.5	44.7	49.5	51.0
SymbCoT (Xu et al., 2024)	56.8	16.5	66.5	75.1	49.5	49.5
LINA (Li et al., 2024a)	48.5	10.8	67.3	69.3	49.5	51.0
HBLR (Li et al., 2025)	54.7	10.4	66.3	62.8	49.0	50.5
MenTaL (Li et al., 2026)	1.8	0.0	75.0	76.2	47.5	50.0

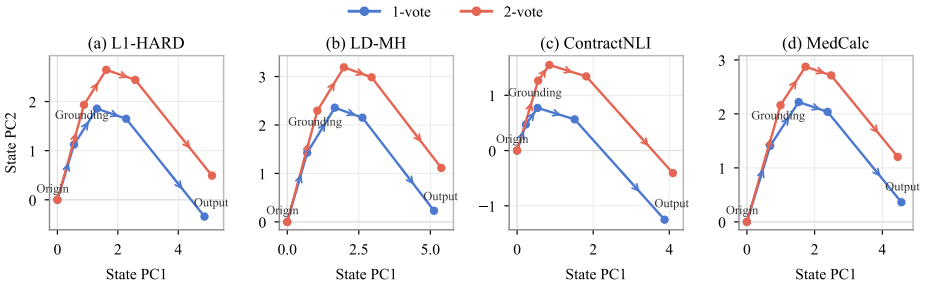


Figure 3: Logged process-state trajectories. Paths join origin, grounding, admission, closure, and output; the first two principal components explain most of the variance in the pooled log features, and ellipses are bootstrap 95% centroid regions. Models and runs are listed in Appendix G.8.

cascade it leads every baseline in every column. The gap between the native and fallback rows is itself the finding, and it is largest where the task asks the model to compose several rules or facts before answering.

On the L1-HARD mid-link-break negatives, chain-of-thought, retrieval, and Logic-LM all stay below eighty percent even when the reasoning budget is tripled. The certified channel serves essentially the entire pool at near-perfect accuracy (McNemar $p<10^{-5}$; full paired tests in Table 11), and the twelve baseline failures we audit consistently omit exactly one intermediate link, which is what a mid-link-break negative is designed to expose.

The pattern repeats on the AFP dependency DAG. Chain-of-thought and retrieval hover around chance because the answer is only well-defined relative to a chain of prior lemmas, while CPUNeSy reconstructs that chain from the stored source and serves the majority of queries at high accuracy.

MedCalc shows a different split that clarifies what execution contributes. The certified channel is highly precise on the questions it answers, but the medical rubric forces frequent abstention when extracted parameters disagree. Routing those abstentions to the same model's open-book answer lifts full-pool accuracy from 72.5% to 90.1% for Seed and from 63.5% to 90.4% for DeepSeek without touching the certified subset, and analogous same-model cascades give a milder but consistent benefit on ContractNLI and LD-MH.

On LeanDojo Benchmark 4, the kernel pool and BM25 both attain seventy-six percent full gold-premise coverage, roughly an order of magnitude above the learned ReProver retriever (Yang et al., 2023). The kernel pool therefore functions as a competitive premise source in its own right. Appendix F gives full protocol details, exact counts, and provenance.

4.2 RQ2: Which Component Recovers Accuracy, and What Does Certification Add?

RQ2 separates deterministic execution from the additional controls on model writes and serving. Execution accounts for most of the accuracy recovery on the controlled derivation tasks, while the added controls change which answers are served, trading coverage for lower accepted-answer risk. On L1-HARD, the uncertified one-vote solver reaches 98.8% full-pool accuracy and the restricted two-vote policy 97.1% (Table 3). Execution alone therefore recovers most of the gain over neural baselines, and the paired difference is not statistically significant ($p = 0.34$; the interval does not establish equivalence either, Appendix G). We therefore treat certification as a selective-serving control rather than an accuracy mechanism. Its value is the error–abstention tradeoff it imposes on top of execution and voting. Because this comparison changes the number of grounding calls, the fixed-trace experiment below isolates the additional effect of source rechecking.

Table 3: L1-HARD ablation (%).

Method	Full	Break
Pure	84.6	69.2
RAG	87.9	75.8
Logic-LM	83.3	68.3
Solver	**98.8**	**99.2**
CPUNeSy	97.1	98.3

We hold both ContractNLI grounding traces fixed and compare plain two-vote agreement with agreement plus source rechecking. For DeepSeek, rechecking withholds $63/252 = 25\%$ of the wrong answers that survive agreement and 51 correct answers. For Seed, it withholds $8/128 = 6.25\%$ of such wrong answers and 14 correct answers. This isolates an error–abstention tradeoff beyond voting. Separately, the restricted L1-HARD interface has no observed wrong answers among 233 served queries; its historical runs and the explicit admission replay have different implementation scopes (Table 7). The logged state differences in Figures 3 and 4 are descriptive, and the attribution to rechecking rests on the paired outcomes rather than those projections. Appendix G reports the complete counts.

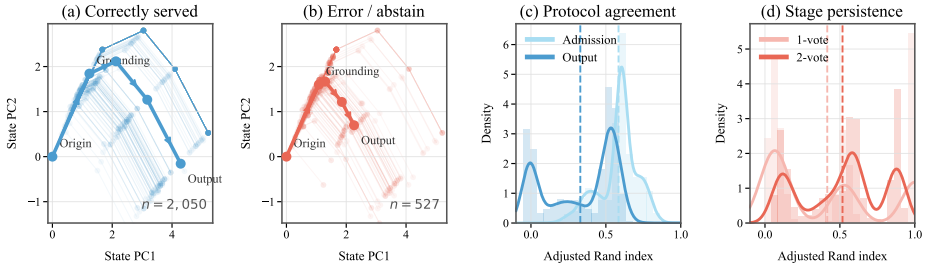


Figure 4: State-cluster consistency across domains. (a–b) Two-vote trajectories for correctly served versus other queries. (c–d) Adjusted Rand index (ARI), with paired bootstrap intervals, measures agreement between protocol clusters and their persistence across stages.

4.3 RQ3: Which Observed Conditions Explain Utility?

RQ1 and RQ2 motivate three diagnostics for interpreting utility. *Derivation demand* counts required rule applications; *kernel adequacy* asks whether the stored source supports a query; and the *grounding-error pattern* describes errors shared across calls versus disagreements. We analyze these properties retrospectively on the recorded runs. The large gains on L1-HARD, M-HARD, and LD-MH are consistent with the role of derivation and source coverage, while the public-benchmark results show that checks can reduce wrong answers without improving full-pool accuracy.

Table 4: CPUNeSy by grounding regime (L1-HARD, %).

Grounder	regime	full	cov.	neural	Δ_{full}
K3	slip	97.1	97.1	87.9	+9.2
K2.6	slip	94.2	94.6	88.3	+5.9
GLM	slip	92.9	93.8	92.9	0.0
DeepSeek	bias	76.7	91.7	84.6	−7.9
Qwen2.5-72B	bias	82.1	93.3	90.0	−7.9

Grouping the L1-HARD runs by their observed joint-error patterns gives a clean split (Table 4). Relative to each model's better neural control, full-pool accuracy increases for K3 and K2.6, is unchanged for GLM, and decreases for DeepSeek and Qwen. This pattern is consistent with the

shared-error limitation in Lemma 1; it does not identify the lemma's latent parameters or establish independence from two calls. Agreement can retain shared errors, which is what the fixed-trace source-recheck comparison in RQ2 measures. Table 14 gives the full cross-model results. Correct answers, errors, and abstentions also distribute differently over vote disagreement and grounding coverage (Figure 5), and the overlap on ContractNLI illustrates why these two coordinates alone do not establish source support. These observations motivate a calibration protocol.

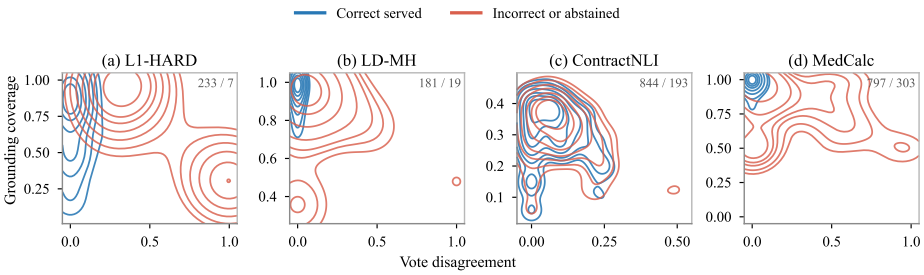


Figure 5: Vote-level grounding geometry (K3 for L1-HARD/LD-MH; Seed for ContractNLI/MedCalc). The LD-MH panel uses a different model from the Seed column in Table 2. The horizontal axis shows vote disagreement (one minus Jaccard) and the vertical axis grounding coverage. Blue marks correctly answered queries and coral marks errors or abstentions; contours are normalized within each class.

4.4 RQ4: What Is the Serving-Time Cost of Controlling Model Writes?

The resource ledger distinguishes candidate-pool restriction, closure-preserving kernelization, and selective serving. On M-HARD, restricting the candidate vocabulary to leaf-Ω retains 48.9% of full-pool units at the same observed 97.45% answered accuracy and reduces tokens by 43% (Table 7). These are candidate-restriction measurements rather than a measured storage-byte saving from the deletion scan. The separate corpus audit finds bounded rederivation redundancy of 37.1% for AFP and 0.60% for law, both below the protocol's $\tau = 0.5$ compression-yield target (Table 6). Physical storage and the scheduler-specific crossover remain unmeasured.

Repeated grounding and rechecking trade wrong answers against abstentions and compute, and their utility depends on all three costs. Under the measured token ledger, restricted two-call serving on L1-HARD is preferred when $2c_W > 6c_A + 145{,}422c_T$ over the 240 items, with c_W, c_A, and c_T pricing a wrong answer, an abstention, and a token (derivation in Appendix G). Counts and answered-risk intervals for each recorded policy, including fallback, appear in Table 10.

5 Conclusion

When a language model is paired with a symbolic executor, two things change at once, and they are easy to confuse. The model stops deriving by sampling and starts deriving by solving, and its premises become gated writes instead of free text. CPUNESY separates the two, pairing a certificate gate that controls what the model may write into symbolic state with a deterministic executor that derives the answer. Execution accounts for most of the accuracy recovery on derivation-heavy tasks, lifting full-pool accuracy on L1-HARD from 87.9% for the strongest neural baseline to 98.8% with no certificate at all. The certificate adds something different. With grounding traces held fixed on ContractNLI, source rechecking removes a quarter of DeepSeek's wrong answers that survive two-vote agreement, at the cost of withholding correct answers it cannot support. The practical reading is that a deployment needing accuracy on derivation-heavy tasks gets most of it from the solver alone, while one whose failures must not be silent needs the gate, at roughly twice the tokens. The boundaries are equally specific. CPUNESY's certificates cover derivational validity relative to admitted premises, so strengthening the method reduces to strengthening the source checker, and where grounding errors are shared across calls, agreement repeats them and the gate cannot help. The next step is prospective calibration, measuring derivation demand, kernel adequacy, and grounding regime on a disjoint annotated set before freezing the serving policy.

Ethics Statement

The legal and medical experiments evaluate benchmark inference and calculation. Clinical or legal use would require further validation of source interpretation, since the model can misread a contract or clinical note even when the subsequent inference passes verification. Applications should expose source evidence, checker identity, and abstention status, and distinguish fallback answers from checked outputs. Decisions affecting patients or legal rights require appropriate expert review.

AI Use Statement

Language models served as experimental subjects and grounding components, and assisted coding, writing, visualization, and offline analysis. Human authors are responsible for verifying all records, results, and claims.

References

Samy Badreddine, Artur d'Avila Garcez, Luciano Serafini, and Michael Spranger. Logic tensor networks. *Artificial Intelligence*, 303:103649, 2022. doi: 10.1016/j.artint.2021.103649.

Tarek R. Besold, Artur d'Avila Garcez, Sebastian Bader, Howard Bowman, Pedro Domingos, Pascal Hitzler, et al. Neural-symbolic learning and reasoning: A survey and interpretation. In *Neuro-Symbolic Artificial Intelligence: The State of the Art*, pp. 1–51. IOS Press, 2022.

Camille Bourgaux, Pierre Bourhis, Liat Peterfreund, and Michaël Thomazo. Revisiting semiring provenance for datalog. In *Proceedings of the 19th International Conference on Principles of Knowledge Representation and Reasoning*, pp. 91–101, 2022. doi: 10.24963/kr.2022/10. URL https://doi.org/10.24963/kr.2022/10.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021. doi: 10.48550/arXiv.2110.14168. URL https://arxiv.org/abs/2110.14168.

William W. Cohen, Fan Yang, and Kathryn Rivard Mazaitis. TensorLog: A probabilistic database implemented using deep-learning infrastructure. *Journal of Artificial Intelligence Research*, 67: 285–325, 2020. doi: 10.1613/jair.1.11944.

Brandon C. Colelough and William Regli. Neuro-symbolic AI in 2024: A systematic review. *arXiv preprint arXiv:2501.05435*, 2025.

Artur S. d'Avila Garcez and Gerson Zaverucha. The connectionist inductive learning and logic programming system. *Applied Intelligence*, 11(1):59–77, 1999.

Luc De Raedt, Angelika Kimmig, and Hannu Toivonen. ProbLog: A probabilistic Prolog and its application in link discovery. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 2462–2467, 2007.

Yixin Dong, Charlie F. Ruan, Yaxing Cai, Ruihang Lai, Ziyi Xu, Yilong Zhao, and Tianqi Chen. XGrammar: Flexible and efficient structured generation engine for large language models. *Proceedings of Machine Learning and Systems*, 7, 2025. URL https://proceedings.mlsys.org/paper_files/paper/2025/hash/5c20ca4b0b20b0bd2f1d839dc605e70f-Abstract-Conference.html.

Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jiang, Bill Yuchen Lin, Sean Welleck, Peter West, Chandra Bhagavatula, Ronan Le Bras, Jena D. Hwang, Soumya Sanyal, Xiang Ren, Allyson Ettinger, Zaid Harchaoui, and Yejin Choi. Faith and fate: Limits of transformers on compositionality. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 70293–70332, 2023.

Richard Evans and Edward Grefenstette. Learning explanatory rules from noisy data. *Journal of Artificial Intelligence Research (JAIR)*, 61:1–64, 2018.

Boris Galitsky, Vladimir Solodkin, and Aleksandr Beznosikov. MetaPCR-LLM: Universal proof-carrying reasoning across heterogeneous logics for large language model validation. *Preprints.org preprint 202609.0849*, 2026. doi: 10.20944/preprints202609.0849.v1. URL https://www.preprints.org/manuscript/202609.0849.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. PAL: Program-aided language models. In *International Conference on Machine Learning (ICML)*, pp. 10764–10799, 2023.

Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 4878–4887, 2017.

Lukas Helff, Ahmad Omar, Felix Friedrich, Antonia Wüst, Hikaru Shindo, Rupert Mitchell, Tim Woydt, Patrick Schramowski, Wolfgang Stammer, and Kristian Kersting. SLR: Automated synthesis for scalable logical reasoning. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 402–426. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.16. URL https://aclanthology.org/2026.acl-long.16/.

Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963.

Albert Q. Jiang, Sean Welleck, Jin Peng Zhou, Timothée Lacroix, Jiacheng Liu, Wenda Li, Mateja Jamnik, Guillaume Lample, and Yuhuai Wu. Draft, sketch, and prove: Guiding formal theorem provers with informal proofs. In *International Conference on Learning Representations (ICLR)*, 2023.

Dongwei Jiang, Marcio Fonseca, and Shay B. Cohen. LeanReasoner: Boosting complex logical reasoning with lean. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 7497–7510. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.naacl-long.416. URL https://aclanthology.org/2024.naacl-long.416/.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020. doi: 10.48550/arXiv.2001.08361. URL https://arxiv.org/abs/2001.08361.

Parisa Kordjamshidi, Samer Aslan, Madhavan Seshadri, Leslie Barrett, and Enrico Santus. Reasoners or translators? contamination-aware evaluation and neuro-symbolic robustness on tax law. In *Proceedings of the First Workshop on Structured Understanding, Retrieval, and Generation in the LLM Era (SURGeLLM 2026)*, pp. 344–360. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.surgellm-1.23. URL https://aclanthology.org/2026.surgellm-1.23/.

Guillaume Lample, Timothée Lacroix, Marie-Anne Lachaux, Aurélien Rodriguez, Amaury Hayat, Thibaut Lavril, Gabriel Ebner, and Xavier Martinet. HyperTree proof search for neural theorem proving. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 26337–26349, 2022.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 9459–9474, 2020.

Qingchuan Li, Jiatong Li, Tongxuan Liu, Yuting Zeng, Mingyue Cheng, Weizhe Huang, and Qi Liu. Leveraging LLMs for hypothetical deduction in logical inference: A neuro-symbolic approach. *arXiv preprint arXiv:2410.21779*, 2024a. URL https://arxiv.org/abs/2410.21779.

Qingchuan Li, Mingyue Cheng, Zirui Liu, Daoyu Wang, Yuting Zeng, and Tongxuan Liu. From hypothesis to premises: LLM-based backward logical reasoning with selective symbolic translation. *arXiv preprint arXiv:2512.03360*, 2025. URL https://arxiv.org/abs/2512.03360.

Qingchuan Li, Jiatong Li, Zirui Liu, Mingyue Cheng, Yuting Zeng, Qi Liu, and Tongxuan Liu. Are llms stable formal logic translators in logical reasoning across linguistically diversified texts?, 2026. URL https://arxiv.org/abs/2506.04575. Version 3; accepted by The Web Conference 2026.

Zhaoyu Li, Jialiang Sun, Logan Murphy, Qidong Su, Zenan Li, Xian Zhang, Kaiyu Yang, and Xujie Si. A survey on deep learning for theorem proving. *arXiv preprint arXiv:2404.09939*, 2024b. doi: 10.48550/arXiv.2404.09939. URL https://arxiv.org/abs/2404.09939.

Ziyang Li, Jiani Huang, and Mayur Naik. Scallop: A language for neurosymbolic programming. *Proceedings of the ACM on Programming Languages*, 7(PLDI):1463–1487, 2023. doi: 10.1145/3591280. URL https://doi.org/10.1145/3591280.

Chengwu Liu, Ye Yuan, Yichun Yin, Yan Xu, Xin Xu, Zaoyu Chen, Yasheng Wang, Lifeng Shang, Qun Liu, and Ming Zhang. Safe: Enhancing mathematical reasoning in large language models via retrospective step-aware formal verification. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 12171–12186, 2025. doi: 10.18653/v1/2025.acl-long.594. URL https://aclanthology.org/2025.acl-long.594/.

Robin Manhaeve, Sebastijan Dumančić, Angelika Kimmig, Thomas Demeester, and Luc De Raedt. DeepProbLog: Neural probabilistic logic programming. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/hash/dc5d637ed5e62c36ecb73b654b05ba2a-Abstract.html.

Emanuele Marconato, Stefano Teso, Antonio Vergari, and Andrea Passerini. Not all neuro-symbolic concepts are created equal: Analysis and mitigation of reasoning shortcuts. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 72507–72539, 2023. doi: 10.52202/075280-3170.

Maciej Mikuła, Szymon Tworkowski, Szymon Antoniak, Bartosz Piotrowski, Albert Qiaochu Jiang, Jin Peng Zhou, Christian Szegedy, Łukasz Kuciński, Piotr Miłoś, and Yuhuai Wu. MagnusHammer: A transformer-based approach to premise selection. In *International Conference on Learning Representations (ICLR)*, 2024.

Theo Olausson, Alex Gu, Ben Lipkin, Cedegao Zhang, Armando Solar-Lezama, Joshua Tenenbaum, and Roger Levy. LINC: A neurosymbolic approach for logical reasoning by combining language models with first-order logic provers. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 5153–5176. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.313. URL https://aclanthology.org/2023.emnlp-main.313/.

Liangming Pan, Alon Albalak, Xinyi Wang, and William Yang Wang. Logic-LM: Empowering large language models with symbolic solvers for faithful logical reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 3806–3824, 2023. doi: 10.18653/v1/2023.findings-emnlp.248.

Mihir Parmar, Nisarg Patel, Neeraj Varshney, Mutsumi Nakamura, Man Luo, Santosh Mashetty, Arindam Mitra, and Chitta Baral. LogicBench: Towards systematic evaluation of logical reasoning ability of large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13679–13707. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.739. URL https://aclanthology.org/2024.acl-long.739/.

Stanislas Polu and Ilya Sutskever. Generative language modeling for automated theorem proving. *arXiv preprint arXiv:2009.03393*, 2020. doi: 10.48550/arXiv.2009.03393. URL https://arxiv.org/abs/2009.03393.

Chengwen Qi, Ren Ma, Bowen Li, He Du, Binyuan Hui, Jinwang Wu, Yuanjun Laili, and Conghui He. Large language models meet symbolic provers for logical reasoning evaluation. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=C25SgeXWjE.

Siyuan Qiu and Jianfeng Xu. Research on a general state formalization method from the perspective of logic. *Mathematics*, 13(20):3324, 2025. doi: 10.3390/math13203324.

Matthew Richardson and Pedro Domingos. Markov logic networks. *Machine Learning*, 62(1–2): 107–136, 2006.

Ryan Riegel, Alexander Gray, Francois Luus, Naweed Khan, Ndivhuwo Makondo, Ismail Yunus Akhalwaya, Haifeng Qian, Ronald Fagin, Francisco Barahona, Udit Sharma, Shajith Ikbal, Hima Karanam, Sumit Neelam, Ankita Likhyani, and Santosh Srivastava. Logical neural networks. *arXiv preprint arXiv:2006.13155*, 2020. doi: 10.48550/arXiv.2006.13155. URL https://arxiv.org/abs/2006.13155.

Tim Rocktäschel and Sebastian Riedel. End-to-end differentiable proving. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/hash/b2ab001909a8a6f04b51920306046ce5-Abstract.html.

Hyun Ryu, Gyeongman Kim, Hyemin S. Lee, and Eunho Yang. Divide and translate: Compositional first-order logic translation and verification for complex logical reasoning. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=09FiNmvNMw.

Torsten Scholak, Nathan Schucher, and Dzmitry Bahdanau. PICARD: Parsing incrementally for constrained auto-regressive decoding from language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 9895–9901. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.emnlp-main.779. URL https://aclanthology.org/2021.emnlp-main.779/.

Bernhard Scholz, Herbert Jordan, Pavle Subotić, and Till Westmann. On fast large-scale program analysis in Datalog. In *Proceedings of the International Conference on Compiler Construction (CC)*, pp. 196–206, 2016. doi: 10.1145/2892208.2892226.

Arseny Skryagin, Wolfgang Stammer, Daniel Ochs, Devendra Singh Dhami, and Kristian Kersting. Neural-probabilistic answer set programming. In *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning (KR)*, pp. 463–473, 2022. doi: 10.24963/kr.2022/48.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R. Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research (TMLR)*, 2023.

Hongda Sun, Weikai Xu, Wei Liu, Jian Luan, Bin Wang, Shuo Shang, Ji-Rong Wen, and Rui Yan. DetermLR: Augmenting LLM-based logical reasoning from indeterminacy to determinacy. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9828–9862, 2024. doi: 10.18653/v1/2024.acl-long.531. URL https://aclanthology.org/2024.acl-long.531/.

Maxim Sviridenko. A note on maximizing a submodular set function subject to a knapsack constraint. *Operations Research Letters*, 32(1):41–43, 2004. doi: 10.1016/S0167-6377(03)00062-2.

Amitayush Thakur, George Tsoukalas, Yeming Wen, Jimmy Xin, and Swarat Chaudhuri. An in-context learning agent for formal theorem-proving. In *First Conference on Language Modeling (COLM)*, 2024. arXiv:2310.04353.

Geoffrey G. Towell and Jude W. Shavlik. Knowledge-based artificial neural networks. *Artificial Intelligence*, 70(1–2):119–165, 1994.

Trieu H. Trinh, Yuhuai Wu, Quoc V. Le, He He, and Thang Luong. Solving olympiad geometry without human demonstrations. *Nature*, 625(7995):476–482, 2024.

Po-Wei Wang, Priya L. Donti, Bryan Wilder, and J. Zico Kolter. SATNet: Bridging deep learning and logical reasoning using a differentiable satisfiability solver. In *International Conference on Machine Learning (ICML)*, pp. 6545–6554, 2019.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 24824–24837, 2022.

Yanrui Wu, Lingling Zhang, Xinyu Zhang, Jiayu Chang, Pengyu Li, Xu Jiang, Jingtao Hu, and Jun Liu. LogicGraph: Benchmarking multi-path logical reasoning via neuro-symbolic generation and verification. *arXiv preprint arXiv:2602.21044*, 2026. doi: 10.48550/arXiv.2602.21044. URL https://arxiv.org/abs/2602.21044.

Jianfeng Xu. Derivation depth as an information metric: Axioms, coding theorems, and storage–computation tradeoffs. *arXiv preprint arXiv:2602.19137*, 2026a. doi: 10.48550/arXiv.2602.19137. URL https://arxiv.org/abs/2602.19137.

Jianfeng Xu. Proof-valid benchmarking for approximate LLM caching under premise erasures. *ChinaXiv preprint ChinaXiv:202608.00158*, 2026b. doi: 10.12074/202608.00158. URL https://chinaxiv.org/abs/202608.00158.

Jianfeng Xu. Proof-valid caching under premise erasures: Local structural limits and shared-workload gains. *arXiv preprint arXiv:2608.11782*, 2026c. doi: 10.48550/arXiv.2608.11782. URL https://arxiv.org/abs/2608.11782.

Jianfeng Xu. Rate-distortion theory for deductive sources under closure fidelity. *arXiv preprint arXiv:2604.15698*, 2026d. doi: 10.48550/arXiv.2604.15698. URL https://arxiv.org/abs/2604.15698.

Jingyi Xu, Zilu Zhang, Tal Friedman, Yitao Liang, and Guy Van den Broeck. A semantic loss function for deep learning with symbolic knowledge. In *International Conference on Machine Learning (ICML)*, pp. 5502–5511, 2018.

Jundong Xu, Hao Fei, Liangming Pan, Qian Liu, Mong-Li Lee, and Wynne Hsu. Faithful logical reasoning via symbolic chain-of-thought. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13326–13365. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.720. URL https://aclanthology.org/2024.acl-long.720/.

Jundong Xu, Hao Fei, Meng Luo, Qian Liu, Liangming Pan, William Yang Wang, Preslav Nakov, Mong-Li Lee, and Wynne Hsu. Aristotle: Mastering logical reasoning with a logic-complete decompose-search-resolve framework. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3052–3075. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.acl-long.153. URL https://aclanthology.org/2025.acl-long.153/.

Kaiyu Yang, Aidan Swope, Alex Gu, Rahul Chalamala, Peiyang Song, Shixing Yu, Saad Godil, Ryan Prenger, and Anima Anandkumar. LeanDojo: Theorem proving with retrieval-augmented language models. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2023.

Sen Yang, Xin Li, Leyang Cui, Lidong Bing, and Wai Lam. Neuro-symbolic integration brings causal and reliable reasoning proofs. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 5732–5744. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-naacl.317. URL https://aclanthology.org/2025.findings-naacl.317/.

Zhun Yang, Adam Ishay, and Joohyung Lee. NeurASP: Embracing neural networks into answer set programming. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 1755–1762, 2020. doi: 10.24963/ijcai.2020/243.

Xi Ye, Qiaochu Chen, Isil Dillig, and Greg Durrett. SatLM: Satisfiability-aided language models using declarative prompting. In *Advances in Neural Information Processing Systems*, volume 36, pp. 45548–45580. Curran Associates, Inc., 2023. doi: 10.52202/075280-1974. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/8e9c7d4a48bdac81a58f983a64aaf42b-Paper-Conference.pdf.

Ruiwen Zhou, Wenyue Hua, Liangming Pan, Sitao Cheng, Xiaobao Wu, En Yu, and William Yang Wang. RuleArena: A benchmark for rule-guided reasoning with LLMs in real-world scenarios. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 550–572. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.acl-long.27. URL https://aclanthology.org/2025.acl-long.27/.

A Full Related Work

This appendix expands the comparison in Section 2, covering neural representations of logic, symbolic execution, premise selection, and source checking.

A.1 First wave: connectionist–symbolic integration (1990s)

KBANN Towell & Shavlik (1994) compiled a propositional Horn rule set into the topology of a feed-forward network and then refined the rules by backpropagation. It supported both rule insertion and extraction from a neural model. CILP d'Avila Garcez & Zaverucha (1999) showed that recurrent networks with semi-linear neurons approximate the fixed-point operator of propositional logic programs, closing a learn–reason–extract cycle with sound extraction Besold et al. (2022). These systems established ways to encode symbolic rules in neural networks. CPUNeSy builds on the separation between rules and their neural representation, with additional checks on source premises and their storage.

A.2 Statistical relational learning and probabilistic logics (2000s)

Markov logic networks Richardson & Domingos (2006) softened first-order clauses with weights; ProbLog De Raedt et al. (2007) gave logic programs a distribution semantics. These methods represent uncertainty over logical statements. Their use of probabilistic semantics addresses a different aspect of inference from the source-admission checks studied here.

A.3 Differentiable deductive engines (2016–2022)

TensorLog Cohen et al. (2020) compiled Datalog-style inference into differentiable matrix algebra; end-to-end differentiable proving Rocktäschel & Riedel (2017) replaced unification by vector similarity; ∂ILP Evans & Grefenstette (2018) learned rules by gradient descent; DeepProbLog Manhaeve et al. (2018) introduced neural predicates over probabilistic facts; NeurASP Yang et al. (2020) embedded networks into answer-set programs; SLASH Skryagin et al. (2022) combined neural predicates with probabilistic circuits. These works made predicate-level processing *trainable*. The learned predicates connect neural predictions to logical inference. Reasoning shortcuts can arise when those predictions fit the task labels while misrepresenting the intended concepts (Appendix A.7).

A.4 Logic as layers and constraints

Logical Neural Networks Riegel et al. (2020) realize logical gates as neurons with provable bound propagation; Logic Tensor Networks Badreddine et al. (2022) ground fuzzy first-order formulas in tensor computations; SATNet Wang et al. (2019) made MaxSAT a differentiable layer; the semantic loss Xu et al. (2018) moves the constraint into the training objective. These methods incorporate

logic into network computation or training objectives. CPUNeSy uses a separate symbolic state whose updates are checked before execution.

A.5 LLM-ERA HYBRIDS

The closest systems translate model output into a symbolic problem. Logic-LM Pan et al. (2023), LINC Olausson et al. (2023), SatLM Ye et al. (2023), LeanReasoner Jiang et al. (2024), and CLOVER Ryu et al. (2025) all make the model translate natural language into a formal object and then rely on a symbolic executor or verifier. Their target formalisms include logic programming, FOL, SAT/SMT, and Lean. They also differ in the repair and decomposition performed before execution. SymbCoT Xu et al. (2024) and Aristotle Xu et al. (2025) keep more of the symbolic reasoning trajectory inside LLM prompting, while CaRing Yang et al. (2025) extracts human-readable proofs from Prolog search logs. These systems motivate the distinction between validating an inference and checking its premises. CPUNeSy admits source units through a certificate gate and records the dependencies of conclusions produced by the symbolic core.

Structured generation addresses errors in output syntax. PICARD Scholak et al. (2021) and XGrammar Dong et al. (2025) constrain decoding so generated strings remain syntactically valid. This is useful for reducing parser failures, but a well-formed formula may still assert an unsupported fact. Datalog engines such as Soufflé Scholz et al. (2016) and provenance semantics for Datalog Bourgaux et al. (2022) provide execution and dependency tracking. CPUNeSy combines these operations with source admission and removal of redundant stored units. Proposition 3 specifies when to cache a derived result under the stated cost model, and Section 4.4 reports the measured costs.

A.6 NEURAL THEOREM PROVING AND PREMISE SELECTION

GPT-f Polu & Sutskever (2020) introduced transformer step-proving on Metamath; HTPS Lample et al. (2022) added search-guided expert iteration; DSP Jiang et al. (2023) pipelines informal drafts to formal sketches; LeanDojo/ReProver Yang et al. (2023) and MagnusHammer Mikuła et al. (2024) made learned premise selection the standard interface to large libraries; AlphaGeometry Trinh et al. (2024) demonstrated a neuro-symbolic loop at olympiad level; Li et al. (2024b) surveys the area. Premise-selection indexes and our kernel store both supply candidates for proof construction. We compare candidate coverage and selection accuracy against ReProver in Appendix F.2.

A.7 FAILURE MODES: REASONING SHORTCUTS

Marconato et al. Marconato et al. (2023) showed that NeSy predictors can reach high label accuracy while violating the intended semantics; mitigation requires extra supervision or architectural constraints. These findings motivate checking which premises support a prediction. Our experiments measure the effects of restricted writes, agreement, and source rechecking (Section 4.2).

A.8 LOGICAL-REASONING BENCHMARKS AND PRESSURE TESTS

Recent benchmarks sharpen the empirical boundary for LLM reasoning. LogicBench Parmar et al. (2024) isolates individual inference rules across propositional, first-order, and non-monotonic patterns; ProverGen/ProverQA Qi et al. (2025) combines LLM generation with symbolic provers to create FOL reasoning problems with verified intermediate chains; RuleArena Zhou et al. (2025) moves rule following into realistic airline, sports, and tax-policy settings. These benchmarks test whether models follow contextual rules through multiple inference steps. Our L1-HARD, M-HARD, ContractNLI, MedCalc, and LeanDojo evaluations examine how deterministic execution and checks on model-proposed premises affect errors, abstentions, and cost.

A.9 POSITIONING

Table 5 identifies the role of related techniques in the present design and pairs each contribution with the evaluation question studied here.

CPUNeSy uses deterministic Horn execution and proof/provenance ideas (Scholz et al., 2016; Polu & Sutskever, 2020; Jiang et al., 2023; Yang et al., 2023), and retains the stated cost assumptions

Table 5: Relationship to established components and evaluation questions.

Family	Established contribution	Question tested here
Parse-and-solve	Execute model-generated formalizations	Which agreed outputs does source rechecking withhold?
Self-consistency	Aggregate repeated predictions	What changes when the two traces and votes are held fixed?
Structured decoding	Enforce syntactic output contracts	Can an allowed semantic assertion still be wrong?
Proof/provenance systems	Check inference and trace dependencies	Which premises may enter the checked state?
Selective prediction	Trade coverage against answered risk	When do error, refusal, and call costs favor checking?
Kernel and cache methods	Reduce stored or recomputed state	Does this corpus offer sufficient redundancy to justify compression?

of the depth–information model (Xu, 2026a). Its primary empirical addition is the explicit decomposition of restricted writes, agreement, and source rechecking with observable error and refusal counts.

B Background, Definitions, and Supporting Results

This appendix fixes the notation and states the background results invoked by CPUNeSy. Section 3.2 states the main guarantees, and Appendix D gives their proofs.

B.1 States, Formulas, and Deductive Closure

Following the Objective Information Theory (OIT) foundation Qiu & Xu (2025), the *state* of an object or system at a time point (or interval) is the semantic valuation of a set of well-formed formulas over a given domain and interpretation. The framework relates a *semantic domain* S_O of statements to a *carrier domain* S_C that represents them Xu (2026a). In CPUNeSy, the language model is the carrier and the symbolic core operates on the statements it proposes.

Definition 1 (Statement universe and proof system Xu (2026d))**.** *Fix a finite ambient universe $\mathbb{S}_O$ of candidate statements and an effective proof system* PS *such that for every finite $\Gamma \subseteq \mathbb{S}_O$ and every $s \in \mathbb{S}_O$, the judgment $\Gamma \vdash s$ is decidable. The induced* deductive closure *is*

$$\mathrm{Cn}(\Gamma) \;:=\; \{\, s \in \mathbb{S}_O : \Gamma \vdash s \,\},$$

which is reflexive ($\Gamma \subseteq \mathrm{Cn}(\Gamma)$), monotone, and idempotent ($\mathrm{Cn}(\mathrm{Cn}(\Gamma)) = \mathrm{Cn}(\Gamma)$), with $\mathrm{Cn}(\Gamma)$ finite for finite Γ.

Assumption 1 (Finite-step closure dynamics Xu (2026d))**.** *There is a computable monotone operator T_{PS} on finite subsets of $\mathbb{S}_O$ with $T_{\mathsf{PS}}(B) \supseteq B$ and $\mathrm{Cn}(B) = \bigcup_{n\geq 0} T^n_{\mathsf{PS}}(B)$, stabilizing in finitely many iterations. Function-free Horn / Datalog fragments satisfy Assumption 1; this is the decidable fragment in which CPUNeSy cores operate (criterion C1 below).*

B.2 Deductive Sources and the Irredundant Kernel

Definition 2 (Deductive source Xu (2026d))**.** *A* deductive source *is a pair (S_O, P_O) where $S_O \subseteq \mathbb{S}_O$ is a finite, effectively listable knowledge base under a fixed canonical order, and P_O is a distribution over S_O. Fidelity is defined by preservation of the statements that can be derived.*

Definition 3 (Kernel decomposition Xu (2026d))**.** *Fixing* PS *and a canonical scan order, a deterministic deletion scan decomposes*

$$S_O \;=\; A \uplus J,$$

where $A = \mathrm{A}(S_O)$ is the irredundant kernel *and J the* redundant part*: every $j \in J$ satisfies $j \in \mathrm{Cn}(S_O \setminus \{j\})$. Under the order-robustness condition $A = \mathrm{Ess}(S_O) := \{s \in S_O : s \notin \mathrm{Cn}(S_O \setminus \{s\})\}$, the kernel is order-invariant. We write $P_A := P_O(A)$ for the* kernel mass *and π_A for the source conditioned on A; the* redundancy fraction *is $1 - P_A$. We use* weighted *kernel mass $P_A = P_O(A)$ for the service distribution and* uniform *kernel mass $P^u_A := |A|/|S_O|$ for the unit count.*

Theorem 2 (Kernel factorization and zero-distortion rate Xu (2026d)). *Let $\hat{S}_O \subseteq \mathrm{Cn}(S_O) \cap \mathbb{S}_O$ (reconstructions stay inside the closure). Then for every $D \geq 0$,*

$$R_{\mathrm{sem}}(D) \;=\; P_A \cdot R^{(A)}\Big(\tfrac{D}{P_A}\Big),$$

i.e. the redundant part is invisible to both rate and distortion. In the nonconfusable (disjoint-kernel) regime, $R_{\mathrm{sem}}(0) = P_A\, H(\pi_A)$; in the general confusable case, $R_{\mathrm{sem}}(0) = P_A\, H_{\Gamma_0}(\pi_A)$ where H_{Γ_0} is the entropy of the kernel confusability hypergraph. The corresponding nonconfusable per-message specialization used by CPUNeSy is stated in Theorem 5.

Theorem 2 relates the storage requirement to the kernel mass under closure fidelity.

B.3 Derivation Depth as an Information Metric

Definition 4 (Derivation depth Xu (2026a)). *For a finite, effectively decidable premise base $B \subseteq \mathbb{S}_O$ with $m := |B|$, let $P_O(s)$ denote the (finite, computable, well-founded) immediate-predecessor set of s. The* derivation depth $D_{\mathrm{d}}(s \mid B)$ *is the length of a shortest derivation of s from B along P_O; it is a unique, finite, computable non-negative integer.*

Theorem 3 (Depth as information Xu (2026a)). *Under a richness condition and generic incompressibility, for information-rich queries with $d := D_{\mathrm{d}}(q \mid B) \geq 1$,*

$$D_{\mathrm{d}}(q \mid B) \;=\; \widetilde{\Theta}\left(\frac{K\big(q \,\big|\, \langle B\rangle\big)}{\log\big(m + D_{\mathrm{d}}(q \mid B)\big)}\right),$$

where $K(\cdot \mid \langle B\rangle)$ is conditional algorithmic description length and $\widetilde{\Theta}$ hides an unavoidable logarithmic addressing overhead. The characterization is tight, and derivation depth coincides with conditional Bennett logical depth under efficient proof simulation.

From depth to caching. The store–compute model of Xu (2026a) supplies the cost scaling used in Proposition 3. The proof of its caching threshold and allocation guarantee appears in Appendix D.8. The reconstruction-depth bound used after pruning is stated in Lemma 3.

Theorem 3 provides the background depth–information characterization; Proposition 3 uses the stationary cost model to decide when recomputation gives way to caching.

B.4 Proof-Valid Reliability Under Premise Erasure

Definition 5 (Premise erasures and transparent caches Xu (2026c)). *Base leaves are independently unavailable with rate ε due to storage loss. Their source support remains valid. A fixed cache $X \subseteq \mathrm{Cn}(B)$ remains available during the erasure trial. Recovery is evaluated on a fixed hereditary witness DAG, with each inference requiring all its designated parents. Only this witness is used for recovery. A leaf is residually exposed if some path from it to q avoids X. Write $D_{\exp}(q; X)$ for these leaves and $D_{\exp}(q) := D_{\exp}(q; \varnothing)$.*

The residual-leaf law is stated in Corollary 1, and proved in Appendix D.9. The following background result records the additional module and complexity claims.

Theorem 4 (Module gains and cache-selection complexity Xu (2026c;b)). *For shared workloads,* semantic modules *yield exact reliability laws under joint and maximal-error criteria; leaf-only transparent storage incurs a first-order overhead factor $1/\varepsilon$, reduced in the saturated shared regime to $\rho/(s\varepsilon)$, where s premises are protected by one module and ρ is the module-to-leaf cost ratio. Optimal selection in general derivation DAGs is NP-complete already at depth two; MDS parity caching is optimal up to one packet against the coded benchmark.*

B.5 Verification of Model-Proposed Derivations

The preceding background concerns deductive sources. The definitions below specify how a carrier trace is mapped to a certified source. Voting is stated in Section 3.2; coverage, certified depth, and scheduling appear in Appendix C. This subsection supplies their common setup.

Definition 6 (Bridge channel model)**.** *The carrier–core interface is modeled by three error channels: (i)* proposal unreliability μ*: each step of the intended derivation is corrupted or missing in the carrier trace with probability at most* μ*; (ii)* verification miss rate η*: a valid step fails verification with probability at most* η*; (iii)* formalization-failure rate ε_{fr}*: each exposed premise fails formalization with probability at most* ε_{fr}*. Channel (iii) consists of an interface-dependent bias component and an i.i.d. slip component (Definition 8), and the independence of channel events (Assumption 2, S3) applies to the slip component only.*

Definition 7 (Grounding interface)**.** *A* grounding interface specification *is a triple* $\mathcal{I} = (\Sigma, \mathcal{L}, \Omega)$*, where* Σ *is a scoped predicate vocabulary with explicit signatures (arity and argument roles),* $\mathcal{L}$ *is a role legend, i.e. a partial injective map from constants to case/entity roles, and* $\Omega \subseteq \Sigma$ *is a declared source-predicate whitelist whose ground instances the carrier may propose directly. The whitelist determines which predicates the model may propose. Their instances still require source checking. A* grounding operator $\tau_{\mathcal{I}}$*, implemented by the carrier, maps each query* x *to a set* $\hat{G}(x)$ *of ground atoms over* Σ*;* $\tau_{\mathcal{I}}$ *implements the vocabulary and role constraints specified by* $\mathcal{I}$*.*

Definition 8 (Bias–slip decomposition)**.** *Fix an interface* $\mathcal{I}$*, a query distribution, and a gold grounding* $G(x)$ *per query. The* i*-th carrier call produces* $\hat{G}_i(x) = G(x) \oplus B_{\mathcal{I}}(x) \oplus S_i(x)$ *(symmetric difference of ground-atom sets), where the* bias *component* $B_{\mathcal{I}}(x)$ *is determined by* $\mathcal{I}$ *and hence identical across calls, and the* slip *components* $S_i(x)$ *are i.i.d. across calls. A component is* answer-critical *for* x *if it changes the answer derived from the grounded premises. Write* $\beta_{\mathrm{ans}}(\mathcal{I}) := \Pr_x[B_{\mathcal{I}}(x)$ *is answer-critical*$]$ *and* $\sigma_c := \Pr_x[S_i(x)$ *is answer-critical*$]$*. The formalization-failure rate of Definition 6(iii) satisfies* $\varepsilon_{fr}(\mathcal{I}) \leq \beta_{\mathrm{ans}}(\mathcal{I}) + \sigma_c$*, with equality on disjoint error events; the bias component is perfectly correlated across traces, the slip component independent (S3).*

Definition 9 (Certified deductive rounding)**.** *Given premise base* B*, a trace* tr*, an edge extractor* E*, and a verifier* V *for a decidable fragment, the verification procedure, called* certified rounding *and denoted* $\mathcal{R}_{E,V}(B, \mathrm{tr})$*, deletes every unverifiable edge, discards conclusions without a complete verified derivation from* B*, and returns the certified source* $(S_{\mathrm{cert}}, \mathsf{PS}_{\mathrm{cert}})$*. A* certificate gate *admits into the core only units whose derivation edges all verify.*

Definition 10 (Certified depth; recovery depth)**.** $D_{\mathrm{d}}^{\mathrm{cert}}(q \mid B)$ *is the minimum derivation depth of* q *over derivations whose every edge verifies under* V*. Letting* π *be a minimum-depth derivation of* q*, the* recovery depth $d_{\mathrm{rec}}(q)$ *is the maximum, over edges of* π *pruned by rounding, of the certified depth needed to re-derive the pruned subgoal from* B*.*

Definition 11 (Canonical cache key)**.** *Under the canonical order of Definition 3, every predicate unit* u *has a unique normal form;* $\mathrm{key}(u)$ *is its serialization. Units colliding under* key *are logically equivalent, so the key is a sound cache index for certified content.*

Lemma 2 (Certified rounding yields a deductive source)**.** *For a finite carrier trace and a sound, decidable fragment verifier, whenever the rounding operator of Definition 9 retains a nonempty set of units, its output* $(S_{\mathrm{cert}}, \mathsf{PS}_{\mathrm{cert}})$ *is a deductive source in the sense of Definition 2.*

The proof is given in Appendix D.3. If no unit survives verification, the gate rejects the proposal.

Connection to the main guarantees. Proposition 2 gives the coverage guarantee, Lemma 3 bounds certification overhead, Lemma 1 separates shared bias from independent slips, and Proposition 3 governs caching. Their proofs appear in Appendices D.4, D.5, D.6, and D.8, respectively.

Remark 1 (Observed joint-error diagnostic)**.** *The joint outcomes describe how errors co-occur across calls, the behavior considered in Lemma 1. Broad-interface L1-HARD has* 9 *both-wrong pairs; the restricted interface has none and four correct/wrong pairs. Two calls cannot identify* β_{ans} *and* σ_c *separately or establish independence. The restricted vocabulary blocks direct conclusion proposals, while false allowed leaf assertions remain possible (Appendix G).*

The interface and rounding operator determine what can enter the certified representation. Section 3.2 states the guarantees for source admission and certified derivation, together with the voting bound. Appendix C gives the coverage, kernel-storage, caching, and recovery guarantees.

B.6 Applicability Criteria

Definition 12 (Four-criterion compression applicability)**.** *A knowledge base* $\mathcal{K}$ *is* eligible for the proposed compression-and-caching regime *iff: (C1)* $\mathcal{K}$ *is formalizable in a decidable fragment (e.g.*

function-free Horn/Datalog); (C2) task fidelity is defined by the consequences entailed by the stored statements; (C3) the measured redundancy fraction satisfies $1-P_A \geq \tau$ for a protocol-fixed threshold τ; (C4) the derivation-depth distribution admits a budget compatible with finite-step closure, with query frequencies spanning the f_c threshold so the admission law is operative. The supporting results are Lemma 2 for the certified source, Theorem 5 for kernel storage, and Proposition 3 for scheduling. The criteria are evaluated by the protocols in Appendix E. C3 specifies the target compression yield. Checked execution and closure preservation also apply below this threshold.

C Coverage, Kernel, and Resource Guarantees

The following results describe query coverage, kernel storage, caching, and recovery after storage loss. Each result states its assumptions. Appendix D.10 identifies the rate–distortion and scheduling models used in the proofs.

Proposition 2 (Certified coverage)**.** *Condition on the absence of interface-wide grounding bias. Suppose proposal loss, rejection of a valid edge, and premise-formalization failure occur independently in the slip channel with probabilities at most μ, η, and ε_{fr}. If a query requires n derivation edges and e exposed source premises, then*

$$\Pr[q \text{ has a complete certified derivation}] \;\geq\; (1-\mu)^n(1-\eta)^n(1-\varepsilon_{fr})^e.$$

The proof appears in Appendix D.4. This bound formalizes the derivation-demand condition: local failures compound with path length, while correlated grounding bias is handled separately below.

Lemma 3 (Certified-depth overhead)**.** *For any query admitting a certified reconstruction,*

$$D_{\mathrm{d}}(q \mid B) \leq D_{\mathrm{d}}^{\mathrm{cert}}(q \mid B) \leq D_{\mathrm{d}}(q \mid B) + d_{\mathrm{rec}}(q),$$

where $d_{\mathrm{rec}}(q)$ is the largest additional depth needed to reconstruct a sub-derivation removed by certification.

The proof appears in Appendix D.5. The bound gives the additional derivation depth required after verification.

Theorem 5 (Kernel sufficiency under closure fidelity)**.** *Assume a decidable closure operator and a fixed canonical deletion order. Then (i) $\mathrm{Cn}(A) = \mathrm{Cn}(\mathbb{S}_O)$; (ii) every closure-type query answerable from $\mathbb{S}_O$ is answerable from A, with certified depth increasing by at most the re-derivation depth of the removed units; and (iii) storing any $J' \subseteq J$ does not improve the achievable rate–distortion function under closure fidelity. Hence the persistent source store needs $|A| = P_A^u|\mathbb{S}_O|$ units, a fraction P_A^u of the full source. For nonempty A, uniform messages supported on A in the nonconfusable regime have an asymptotic zero-distortion rate of $\log|A|$ nats per message; uniform symbolwise-exact messages over $\mathbb{S}_O$ require $\log|\mathbb{S}_O|$. The two rates apply to the respective message distributions just specified.*

Each deletion removes only a unit derivable from the remainder, which explains why kernelization preserves closure. The proof appears in Appendix D.7. The guarantee applies only to queries covered by the certified source. The fraction $1 - P_A^u$ measures how much symbolic mass can be removed before serving, which the corpus audit reports as criterion C3.

Proposition 3 (Store–recompute threshold)**.** *Under the stationary depth–information cost model of Xu (2026a), with store-to-recompute rate ratio ρ and effective kernel size m_{eff}, the break-even frequency for caching a certified unit satisfies*

$$f_{\mathrm{c}}(u) = \widetilde{\Theta}\big(\rho \log\big(m_{\mathrm{eff}} + D_{\mathrm{d}}^{\mathrm{cert}}(u)\big)\big)\,.$$

Under a storage budget and a normalized monotone depth-reduction objective with diminishing returns, partial-enumeration greedy attains a $1 - 1/e$ approximation (Sviridenko, 2004).

The proof appears in Appendix D.8. The threshold selects cache candidates, and the allocation rule selects a subset within the storage budget. The logarithmic scale is inherited from the cited depth–information model, while the allocation guarantee additionally requires the stated submodularity condition. Kernelization removes units that do not change closure; caching stores selected consequences that do change runtime cost, deep or frequently used ones first.

Corollary 1 (Reliability under premise erasure)**.** *Under the fixed-cache availability model of Definition 5, independent premise erasure at rate $0 < \varepsilon < 1$, and a canonical-witness derivation DAG, a query with ℓ residually exposed leaves remains recoverable with probability $(1-\varepsilon)^{\ell}$. Caching a valid chain segment improves this probability exactly when it removes at least one leaf from the residual exposed set.*

The proof appears in Appendix D.9.

The three accuracy conditions concern derivation demand (Proposition 2), certified-source coverage (Theorem 5), and slip- versus bias-dominated grounding (Lemma 1). Separately, the corpus audit uses criteria C1–C4. On the $14{,}414$-unit AFP corpus and 503-unit regulatory corpus, C1, C2, and C4 pass. For C3, bounded k-step rederivation saturates at 37.1% and 0.60% for $k \leq 4$, below $\tau = 0.5$ (Table 6). These measurements favor retaining a larger source store. In the regulatory corpus, redundancy is concentrated in the rule layer, and the composition law matches the observed corpus-wide value within half a point.

D Proofs and Auxiliary Results

This appendix proves the results in Section 3.2 and Appendix C. Auxiliary results are stated before their proofs. Appendix D.10 lists the background results used in these arguments.

D.1 Standing Assumptions

The following conditions collect the notation used by the auxiliary arguments. Each result specifies which conditions it requires.

Assumption 2.

S1. ***Decidable fragment with a sound and complete verifier.*** *The core's fragment satisfies Assumption 1, and $V(e,\Gamma) = \top$ iff e is a valid inference step over Γ within the fragment.*

S2. ***Fixed canonical order.*** *The canonical scan order of Definition 3 is public and fixed; kernel extraction is deterministic.*

S3. ***Channel independence.*** *The bridge channel errors (proposal failure rate μ, verification miss rate η, formalization failure rate ε_{fr}) are independent across derivation steps and exposed premises. For the formalization channel this applies to the slip component only; the interface-determined bias component is perfectly correlated across traces and is modeled separately (Definition 8, Lemma 1).*

S4. ***Stationary amortized cost model.*** *Stationary request stream; storage rate α; recomputation rate β; $\rho = \alpha/\beta$; storage cost σ normalized to one step-equivalent per unit.*

S5. ***Canonical-witness erasure regime.*** *Designated-parent map, hereditary derivation DAG (Definition 5); premise erasures i.i.d. with rate ε.*

D.2 Certified-State Soundness

Proof of Theorem 1. Write $\Gamma_t = A_t \cup \mathcal{R}_t$. We induct on completed state transitions; served consequences are evaluated against the state at their serving time. Initialization supplies source-support records for every kernel unit and an empty cache, so all three claims hold.

Source admission. Rejected proposals change no certified state. An accepted unit is type-checked and has a logged source-support record. The deletion scan retains a subset of these source-checked units, hence each new kernel premise still has its record. A newly supported premise may add information beyond the previous kernel. Kernel changes trigger cache revalidation. Each retained entry has a complete verified witness over Γ_{t+1} of depth at most δ; all other entries are evicted. Soundness of the fragment verifier proves its conclusion is in $\mathrm{Cn}(\Gamma_{t+1})$, and the minimum certified depth is no greater than the depth of this witness. Thus (i) and (ii) are preserved.

Derived admission and queries. The gate requires a complete acyclic witness rooted at the proposed conclusion, with source leaves in Γ_t. Induction over its topologically ordered, verified inference

steps gives $u \in \mathrm{Cn}(\Gamma_t)$. Its checked witness depth $d \leq \delta$ implies $D_{\mathrm{d}}^{\mathrm{cert}}(u \mid \Gamma_t) \leq d \leq \delta$. Serving or caching this unit changes no source premise. A cache hit reuses a certificate validated for the current kernel version, so the same argument applies. A rejected or uncertified fallback answer is outside the certified serving channel.

Erasure and eviction. Erasing a source identifier removes it from the active certified source set and cascade-evicts every entry whose transitive certificate dependencies include that identifier. This proves (iii), including dependencies through cached intermediate results. Recomputing the kernel and revalidating any surviving cache entries preserves (i) and (ii) by the source-admission argument. Cache eviction removes an entry and any entries depending on it; source eviction follows the erasure rule. Neither operation admits unsupported content. Frequency updates change no certified content. These cases exhaust the transitions and complete the induction. □

Applying the coverage and voting results. For a query whose reconstruction fits the serving budget, Proposition 2 and Lemma 3 supply the coverage and depth guarantees. Voting is governed separately by Lemma 1.

D.3 Certified Rounding as a Deductive Source

Proof of Lemma 2. S_{cert} collects the conclusions of extracted derivation DAGs all of whose edges verify. It is finite, since the carrier trace is finite, and effectively listable, since edge verification is decidable. Its nonemptiness permits the empirical distribution $\mathsf{PS}_{\mathrm{cert}}$ to be normalized. Soundness of V gives $S_{\mathrm{cert}} \subseteq \mathrm{Cn}(B)$. Thus the finite, effectively listable source with its empirical distribution satisfies Definition 2. □

D.4 Certified Coverage

Proof of Proposition 2. Work on the bias-free query mass specified in the proposition. For the intended derivation to survive rounding, it suffices that each of its n steps be proposed, each of those steps pass verification, and each of its e exposed premises be formalized. The respective probabilities are at least $(1-\mu)^n$, $(1-\eta)^n$, and $(1-\varepsilon_{fr})^e$. Independence in the slip channel makes the probability of their intersection at least their product. On this event, rounding retains a complete certified derivation, which proves the claimed lower bound. □

D.5 Certified-Depth Overhead

Proof of Lemma 3. Certified derivations form a subset of all derivations, giving the lower bound. For the upper bound, let π attain $D_{\mathrm{d}}(q \mid B)$. Restore each subgoal removed by rounding using its shortest certified reconstruction from B. Under the recovery-depth accounting of Definition 10, this adds at most $d_{\mathrm{rec}}(q)$ to the derivation depth. The resulting derivation is certified, giving the upper bound. □

D.6 Voting Under Shared Bias and Independent Slips

Proof of Lemma 1. Condition on a query in the shared-error regime, whose total mass is β_{ans}. By the lemma's explicit assumption, indicators that a trace certifies the common wrong answer are conditionally independent with means at least $1 - \sigma_c > 1/2$. Hoeffding's inequality Hoeffding (1963) bounds the probability of failing to obtain a strict majority of that answer by $e^{-2k(1/2-\sigma_c)^2}$. Multiplication by β_{ans} gives the lower bound in the lemma.

For a query outside that regime, let X_i indicate that trace i certifies the unique correct answer. By the lemma's assumptions, the X_i are independent with means at least $1/2 + \delta_0$. Failure to obtain a strict correct majority is contained in $\{\sum_i X_i \leq k/2\}$, whose probability is at most $e^{-2k\delta_0^2}$ by Hoeffding's inequality. Adding the worst-case contribution from the biased query mass gives the stated upper bound after averaging over queries. A missing strict majority yields abstention, including ties. The lower concentration bound requires the stated conditional independence within the shared-error regime. □

D.7 KERNEL SUFFICIENCY AND STORAGE

Proof of Theorem 5. Part (i): closure preservation. The deletion scan processes S_O in the canonical order and removes j only when $j \in \mathrm{Cn}(\Gamma \setminus \{j\})$ for the current set Γ. We claim $\mathrm{Cn}(\Gamma)$ is invariant under each removal. Inclusion $\mathrm{Cn}(\Gamma \setminus \{j\}) \subseteq \mathrm{Cn}(\Gamma)$ is monotonicity. Conversely, $j \in \mathrm{Cn}(\Gamma \setminus \{j\})$ gives $\Gamma \subseteq \mathrm{Cn}(\Gamma \setminus \{j\})$, whence $\mathrm{Cn}(\Gamma) \subseteq \mathrm{Cn}(\mathrm{Cn}(\Gamma \setminus \{j\})) = \mathrm{Cn}(\Gamma \setminus \{j\})$ by monotonicity and idempotence of Cn (Definition 1). Induction over the scan terminates at A, so $\mathrm{Cn}(A) = \mathrm{Cn}(S_O)$.

Part (ii): query preservation. Closure equality gives the same closure-type answers from A and S_O. A derivation using a removed unit can be reconstructed by replacing that unit with its derivation from the retained kernel. The depth increase is accounted for by the re-derivation depth of the removed units involved, as in Lemma 3.

Part (iii): rate–distortion invariance. The closure-fidelity factorization in Theorem 2, imported from Xu (2026d, Theorem 5.1), makes the redundant part invisible to rate and distortion. Adding $J' \subseteq J$ to the stored kernel therefore leaves the achievable rate–distortion function unchanged.

Storage count and the per-message specialization. The unit-count identity follows from $P_A^u = |A|/|S_O|$ in Definition 3. It concerns the physical number of persistent units. For the final specialization, condition messages on A and take the uniform law there. Then $P_A = 1$ and $H(\pi_A) = \log|A|$; Theorem 2 gives the asserted asymptotic rate. Uniform symbolwise-exact coding on $\mathbb{S}_O$ has entropy $\log|\mathbb{S}_O|$. This calculation compares different source laws. For the original source law, the nonconfusable rate remains $P_A H(\pi_A)$, as in Xu (2026d, Theorem 3.1). □

Remark 2 (Scope of the kernel guarantee)**.** *The unit-count consequence of Theorem 5 and its final per-message claim concern different resources. The former counts persistent units, while the latter uses a uniform distribution on the nonconfusable core alphabet.*

Corollary 2 (Redundancy in mixed corpora)**.** *Let a mixed corpus decompose as $U = R \uplus F$, where R is a rule layer (safe Horn clauses with non-empty bodies) and F an instance layer of ground units each carrying at least one constant that occurs in no other unit of U (case-individual constants), with derivability that of the certified fragment (θ-subsumption between rule units; forward chaining on ground units). Then every $f \in F$ is essential, the redundancy status of every rule unit is decided entirely within R, and the uniform-reading redundancy fraction obeys the composition law*

$$1 - P_A^u(U) = \frac{|R|}{|U|}\big(1 - P_A^u(R)\big).$$

Proof of Corollary 2. θ-subsumption compares clauses pairwise, so no ground unit can declare a rule unit redundant; rule redundancy is therefore computed within R alone. A ground unit f is derivable only via a rule instance whose body units are all present; by safety, some body unit of any such instance contains f's head constants, hence contains the case-unique constant. Only f mentions that constant. Since every rule has a non-empty body, induction on derivation length shows that no ground unit mentioning it is derivable without f. Hence $f \in \mathrm{Ess}(U)$, $|\mathrm{Ess}(U)| = |\mathrm{Ess}(R)| + |F|$, and the displayed law follows by counting. □

Remark 3 (Applicability of the case-uniqueness premise)**.** *The case-individual-constant premise holds for judgment-style corpora, where each instance introduces fresh party constants. On multi-case collections with shared parties or corporations it must be re-established per constant class (e.g., by namespacing case-local identifiers); otherwise ground facts about a shared entity can become derivable and leave the essential core.*

Remark 4 (Order sensitivity of the kernel)**.** *Theorem 5 uses a fixed canonical deletion order; its closure-preservation argument does not require order robustness. Equality with the order-robust essential set is a separate property of Definition 3. Appendix E reports a gap of 0 units on the regulatory corpus and $|A \triangle \mathrm{Ess}(S_O)| = 2{,}473$ (17%) on the Isabelle corpus. The theorem's final per-message claim retains its explicit nonconfusability qualification.*

Remark 5 (C3 applies to the rule layer)**.** *Corollary 2 implies that applying C3 to raw mixed corpora scales rule redundancy by $|R|/|U|$. Under the case-uniqueness assumption, the instance facts are essential. The rule-layer C3 measurement therefore applies to the units extracted during* DEPLOY *(Algorithm 2, stages D2–D3). The mixed-corpus audit of Appendix E.4 confirms the law. The measured overall redundancy 2.2% matches the composition prediction (rule-layer redundancy $13.8\% \times$*

rule-layer share $14.7\% \approx 2.0\%$*) within half a percentage point, and the rule-layer figure is corroborated by a certified re-measurement (*12.87% $[9.1, 16.6]$*, gated formalization plus* θ*-subsumption matching).*

D.8 Store–Recompute Threshold and Allocation

Proof of Proposition 3. Use Xu (2026a, Theorem 4.1) with baseline equal to the current certified kernel, $m = m_{\text{eff}}$, and operational depth $d = D_{\text{d}}^{\text{cert}}(u)$. The certified cost model uses that theorem's information-rich, generic regime and depth-range assumptions in addition to stationarity. Comparing amortized storage cost with recomputation gives the stated threshold, with certification write costs included in the storage rate α.

For allocation, write the expected depth reduction as $F(X) = \sum_q \nu(q)[n_q(\varnothing) - n_q(X)]$. Then $F(\varnothing) = 0$, and adding cached units cannot increase the required depth. The assumed diminishing-returns property makes each summand submodular; positive linear combinations preserve submodularity. Applying the cited partial-enumeration greedy guarantee to the resulting budgeted objective gives the claimed approximation ratio for additive storage costs Sviridenko (2004). □

Application to the persistent core. Use the current base size for m_{eff} and the certified depth of the candidate unit. In the bounded-rounding regime $d_{\text{rec}}(u) = O(m_{\text{eff}} + D_{\text{d}}(u \mid B))$, Lemma 3 controls reconstruction overhead. Proposition 3 then supplies the candidate threshold and the budgeted allocation rule.

D.9 Reliability Under Premise Erasure

Proof of Corollary 1. Fix the cache before erasure. In the witness DAG, recovery fails iff a missing leaf has a cache-free path to the root: tracing a failed uncached node backwards finds a failed parent and eventually such a leaf; conversely, a missing leaf propagates failure along any uncached path. This is the local failure-path argument of Xu (2026c, Lemma 3.1 and Theorem 3.1). Thus recovery is precisely the event that every leaf in the fixed set $D_{\exp}(q; X)$ survives. Independence gives $(1-\varepsilon)^{|D_{\exp}(q;X)|}$. Adding a cache segment cannot create an exposed leaf. Since $0 < 1-\varepsilon < 1$, the probability strictly increases iff that set shrinks. □

Scope of the cache model. Corollary 1 models storage loss while the cache survives and source support remains valid. Withdrawing source support invalidates dependent certificates under Theorem 1, so recovery in that setting requires a separate model. Theorem 4 states the shared-module and complexity results.

D.10 Dependencies on Background Results

Remark 6 (Imported ingredients)**.** *The deletion-scan argument is given in Appendix D.7. The rate–distortion and nonconfusable per-message ingredients retain their dependence on Xu (2026d); the scheduling scale retains its dependence on Xu (2026a); and the residual-leaf characterization and module results retain their dependence on Xu (2026c;b). The concentration step uses Hoeffding (1963), and allocation uses the knapsack-constrained submodular result Sviridenko (2004).*

E Measurement Protocols and Domain Instances

The core as a state machine. The active certified source set S_t is indexed by source records; the runtime state contains its kernel facts A_t, rules $\mathcal{R}_t$, certified cache, certificate log, and frequency estimates. Admit(u) uses the source mode of Algorithm 1 and recomputes the kernel. Query(q) checks the derived-mode gate or reuses a certificate validated for the current kernel version. Erase(ℓ) removes the source record and all cache entries with that transitive dependency; the kernel is recomputed and surviving cache certificates revalidated within δ. Evict(u) removes a cache entry and its dependents (source eviction follows Erase); Tick updates frequencies only. These transitions satisfy Theorem 1, proved in Appendix D.2. The insertion guard additionally enforces $\sigma(\text{Cache}_t) \leq \mathcal{B}$; eviction only decreases this storage. The state can both gain and lose units.

E.1 A SERVED QUERY, END TO END

Algorithms 2–3 describe the pipeline. The example below follows one L1-HARD query (family `prelim495x`; story, grounding, and verdict taken from the stored logs). Both grounding votes return the same fact set. The three-rule witness below is reconstructed from those facts and the fixed rules; each step can be checked against the rule set.

Query (story, abridged): Chen and Zhou sign a subscription form (*ren'gou shu*) agreeing to conclude a formal service contract within thirty days; at the deadline Chen refuses without cause. *May Zhou claim damages for breach of the preliminary contract?*

Grounding (bridge, two separately sampled votes; identical both times):

```
subscription_order_or_booking_form(x1)
agrees_future_conclusion(x1)
determinable_parties_and_subject(x1)
refuses_to_conclude_main_contract(p1,x1)
```

Gate: vote 1 = vote 2 permits source and derivation checking; the supported facts and verified steps below permit certification. Disagreement produces an explicit abstention.

Reconstructed certificate (each step checkable against the fixed rule set):

1. `is_preliminary_contract(x1)` ← facts 1–3 [Judicial Interp., art. 6]
2. `fails_conclusion_duty(p1,x1)` ← step 1 + fact 4 [Judicial Interp., art. 7]
3. `may_claim_preliminary_breach_liability(p1,x1)` ← steps 1 + 2 [Civil Code, art. 495]

Served answer: yes (*gold: yes*); derivation depth 3 ≤ budget. This record contains no cache event. **Verbalization** (carrier, non-critical): "Yes. The subscription form constitutes a preliminary contract under Civil Code art. 495, and Chen's unjustified refusal breaches it."

Figure 6: One L1-HARD query with a witness reconstructed from recorded groundings and fixed rules. Identifiers are shortened. The witness was reconstructed offline from the facts and outputs stored by the task adapter.

Algorithm 2 DEPLOY: theory-grounded instance calibration

Require: formalized corpus S_O with rule set $\mathcal{R}$; checker V; canonical order $\prec$; query log or surrogate P_O; hardware cost ratio ρ

Ensure: CPUNeSy instance $(A, \delta, f_c, \mathcal{B})$; eligibility verdict against τ

1: **D1 Fragment Validation**: re-check every unit and rule with V; estimate $\hat{\varepsilon_{fr}}$
2: **D2/D3 Kernel Extraction**: deletion scan over $\prec$ with the certified matcher (Alg. 4, lines 2–3)
3: **D2/D3** (cont.): order-sensitivity report $|A \triangle \mathrm{Ess}(S_O)|$
4: **D4 Criterion Measurement**: estimate $\widehat{1 - P_A}$ (uniform and weighted, with CIs); verdict vs. τ
5: **D5–D7 Instance Configuration**: $\delta \leftarrow$ 95th P_O-percentile of $\widehat{F}_{D_d}$
6: **D5–D7** (cont.): $f_c(u) \leftarrow \rho \log\left(m_{\text{eff}} + D_d^{\text{cert}}(u \mid B)\right)$ for all u
7: **D5–D7** (cont.): $\mathcal{B}$ sized around $|A|$; check the diminishing-returns condition on the dependency DAG
8: **return** instance configuration and criterion verdicts C1–C4

E.2 MEASUREMENT PROTOCOL

Both domains use the batch kernel-extraction procedure in Algorithm 4.

E.3 DOMAIN INSTANCE 1: FORMAL MATHEMATICS

Corpus. A fixed snapshot of the Isabelle/HOL standard library plus selected Archive of Formal Proofs (AFP) entries (alternative: Lean4 mathlib). Statements (theorems, definitions) become predicate units; proof-term dependencies become rule edges: lemma φ proved from lemmas $\psi_1, \ldots, \psi_k$ by a named inference rule yields the Horn clause $\psi_1 \wedge \cdots \wedge \psi_k \rightarrow \varphi$. The fragment is function-free Horn over a finite statement universe: C1 holds by construction. The source libraries were checked by their proof kernels. Our executor performs closure queries on their recorded dependency DAGs.

Algorithm 3 SERVE: online certified query serving

Require: running core $(A_t, \mathcal{R}_t, \mathrm{Cache}_t)$; query q; k carrier traces; depth budget δ
Ensure: certified answer, or explicit abstention with any fallback flagged uncertified

S1 Grounding Bridge: ground the traces via interface $\mathcal{I}$; decline if the required agreement fails
S2 Source Check: check agreed query facts against their source evidence using SOURCE mode of Algorithm 1; scope these facts to the query
S3 Derivation Check: use DERIVED mode to check a complete witness for q over the current checked premises
if the gate returns `CERTIFY` **then**
    S4: serve the certified answer
    S5 Schedule: consider its witness for caching when $f(q) > f_{\mathrm{c}}(q)$
    S6 Allocate: select among eligible witnesses under $\sigma(X) \leq \mathcal{B}$ by partial-enumeration greedy
else
    decline explicitly; label any optional fallback uncertified
end if
S7 Premise Amendment: on source withdrawal or expiry of query-scoped facts, cascade-evict dependents, recompute the kernel, and revalidate surviving cache certificates

Algorithm 4 Batch kernel extraction and criterion measurement

Require: formalized corpus S_O (units + rule set $\mathcal{R}$); canonical order $\prec$; checker V; query log or surrogate frequency model P_O
Ensure: kernel A, estimates $\widehat{1 - P_A}$, depth distribution $\widehat{F}_{D_{\mathrm{d}}}$, criterion verdicts C1–C4

Validate fragment (C1): re-check every corpus edge with V; report the fraction of units that fail verification (formalization-failure estimate $\hat{\varepsilon_{fr}}$)
Deletion scan: process S_O in order $\prec$; mark s redundant iff $s \in \mathrm{Cn}(\Gamma \setminus \{s\})$ for the surviving set Γ (Definition 3)
$A \leftarrow$ surviving set; $J \leftarrow S_O \setminus A$
Order-robustness check: compute $\mathrm{Ess}(S_O) = \{s : s \notin \mathrm{Cn}(S_O \setminus \{s\})\}$; report the symmetric difference $|A \triangle \mathrm{Ess}(S_O)|$
Redundancy estimate (C3): $\widehat{1 - P_A}$ under both readings (uniform, $1 - |A|/|S_O|$, and frequency-weighted, $1 - P_O(A)$), with Wilson 95% CIs over a bootstrap of corpus slices
Depth distribution (C4): compute $D_{\mathrm{d}}(s \mid A)$ for all $s \in \mathrm{Cn}(A)$ within a cap $\delta_{\max}$ via the predecessor operator (Definition 4); report quantiles and the budget δ covering 95% of P_O-mass
Frequency span: verify that the query-frequency distribution straddles f_{c} over the observed depth range
return $(A, \widehat{1 - P_A}, \widehat{F}_{D_{\mathrm{d}}}, \delta)$ and the four verdicts

C2 (closure fidelity). Premise selection, theorem retrieval, and automated proof search depend on entailment. Equivalent formulations of a lemma are interchangeable for this criterion. C2 holds.

C3 (redundancy). Proof libraries contain substantial derivable material (corollaries, specialized instances, alias lemmas). The scan of Algorithm 4 is run per AFP session with the library order as $\prec$; P_O is taken (a) uniform and (b) proportional to downstream dependency counts (how often a lemma is used as a premise), the latter approximating the service distribution. The protocol hypothesized $1 - P_A \geq 0.6$ under both readings. A first kernel-verified run on a 14-entry subset finds $1 - P_A^u \approx 20\%$ (uniform) and 46.7% with a τ-crossing interval (weighted). See Table 6.

C4 (depth and frequency). Derivation depth is the shortest premise-chain length in the dependency DAG; the frequency surrogate is the dependency count. AFP-scale graphs have shallow depth (order 10–10^2) and heavy-tailed usage, so a budget δ at the 95th percentile is affordable and query frequencies span several orders of magnitude around f_{c}. The kernel-level run supports C4, with shallow depth (quantiles $0/2/3/4$, max 10) and heavy-tailed usage (frequency quantiles $1/4/82$) on real proof-term dependencies.

E.4 Domain Instance 2: Regulatory / Legal Compliance Knowledge Base

The regulatory corpus merges the Chinese Contract Book with the full SPC Interpretation on the General Provisions of the Contract Part (Fa Shi [2023] No. 13). It contains 503 Horn units over 200 articles, with no unsafe rules and 110 source predicates requiring interpretation of the text. C1/C2 hold. Repeated provisions account for 3 certified redundancy edges (interpretation arts. 6/8 vs. statute art. 495; art. 53 vs. art. 565), giving $1 - P_A^u = 0.60\%$ (Wilson $[0.20, 1.74]\%$), weighted 1.24%, schema-alias channel 0.60% (combined 1.19%). The merged dependency DAG has 140 internal edges. Depth quantiles remain $0/1/1/1$ (max 2), so the additional edges increase branching while chain length stays short. **C3 fails** ($0.60\% \ll \tau$), and bounded multi-step derivation finds zero additional redundancy at $k \leq 4$ (Appendix E.7). **Mixed-corpus audit (protocol-specified P-M1–P-M3).** Adding 200 civil judgments (15,859 sentence-level units) shows text-level deduplication removes 6.8% while logical kernel extraction removes a further 2.2 points, 91.7% of it instance/weakening redundancy invisible to text and concentrated on the rule layer (13.1–15.4%). The measured reduction falls below P-M2's predicted ≥ 10 points corpus-wide. The composition law (Corollary 2) accounts for this result. Case-specific facts make up 77% of the corpus and are essential under the stated representation assumptions. Corpus-wide savings therefore depend on both rule-layer redundancy and the share of rules in the corpus. We report C3 for the rule layer as well as the full corpus.

E.5 Setting the Threshold τ

The internal protocol sets the threshold using the storage ratio in Theorem 5. By the unit-count consequence of Theorem 5, the unit-count saving of the kernel store is $|S_O|/|A| = 1/P_A^u$ (uniform reading of the kernel mass). CPUNeSy eligibility should require the saving to materially exceed the fixed cost of running the deductive engine and the bridge (the certification surcharge priced into α, Proposition 3). The internal protocol specifies

$$\tau \;=\; 0.5, \qquad \text{i.e. require} \qquad |S_O|/|A| \;\geq\; 2,$$

so that an eligible core at least halves the unit count; domains in the band $1 - P_A \in [\tau/2, \tau)$ are declared *hybrid-eligible* and served with the graceful-degradation protocol (the main-text hybrid band). A sensitivity analysis over $\tau \in \{0.3, 0.5, 0.7\}$ is reported with the results.

E.6 Threats to Validity

- *Order sensitivity.* The scan kernel depends on $\prec$ unless $A = \mathrm{Ess}(S_O)$; Algorithm 4 reports $|A \triangle \mathrm{Ess}(S_O)|$ so that C3 can be interpreted together with its sensitivity to the deletion order.
- *Frequency surrogates / formalization bias.* Both uniform and weighted readings are reported; LLM-assisted formalization duplicates are removed by the gate's redundancy branch itself, and pre-gate statistics are reported.
- *Fragment coverage.* Excluded open-textured provisions are counted; sub-floor coverage is reported as failing C1.

E.7 Bounded-Step Rederivation and Parent-Designation Sensitivity

We test two sources of sensitivity in the measurements above. Multi-step derivation may reveal redundancy missed by a single-step check. The designated-parent convention in S5 may also change the depth statistics used to set δ.

Bounded k-step C3, $k \in \{1, 2, 3, 4\}$. On the legal corpus the check is exact: for each rule unit r, instantiate its body with fresh Skolem constants as temporary facts and run $\leq k$ semi-naive forward-chaining rounds over the remaining units inside the same corpus-order deletion scan as Algorithm 4. A rule r is k-redundant iff its head instance is rederived. If the number of derived facts exceeds the computation cap, the rule is retained, giving a conservative redundancy estimate. On the AFP corpus the certified match edges (ALIAS $=\alpha$-equivalence; INST = one-step instantiation) compose transitively, so a k-hop path to a surviving unit is a legitimate k-step derivation witness, and the scan deletes a unit iff such a witness exists within k hops. Table 6 reports both domains. The legal corpus

Table 6: Bounded k-step rederivation C3 (uniform $1-P_A^u$, Wilson 95%). k=1 reproduces the single-step measurements. Redundancy saturates at k=2 in both domains and remains below $\tau = 0.5$ through k=4.

Domain (n)	k=1	k=2	k=3	k=4
AFP subset (14,414)	19.97 [19.33, 20.63]	37.13 [36.35, 37.92]	37.13	37.13
Legal merged (503)	0.60 [0.20, 1.74]	0.60	0.60	0.60

shows zero additional redundancy through k=4, consistent with its short chains (maximum depth 2). AFP redundancy nearly doubles at k=2 because the scan can compose matching edges, for example to remove an instance of an alias. The saturated rate remains 12.9 points below τ. These results establish the C3 outcome for the tested depth limits.

Designated-parent sensitivity (S5). The depth assignment of Appendix E.2 charges $D_{\mathrm{d}}(i) = 1 + \max_p D_{\mathrm{d}}(p)$ over premise depths; S5's accounting follows one designated premise edge. To test whether the convention matters, we re-drew the designated parent uniformly at random on every multi-premise unit ($18/503 = 3.6\%$ legal; $2{,}497/14{,}414 = 17.3\%$ AFP) and recomputed depths over 2,000 resamples. The 95th uniform percentile of depth is invariant: constant 1 (legal) and constant 2 (AFP), standard deviation 0.0 in both domains; the max-parent rule is the conservative upper envelope (AFP 3 vs. 2). The depth statistics behind δ and behind C4's long-tail finding are therefore insensitive to S5's parent designation.

F Full Experimental Details

The experiments measure corpus redundancy, derivation depth, grounding errors, and task performance against neural and hybrid baselines. Table 7 summarizes the results for each hypothesis in the internal protocol and identifies measurements still needed.

Table 7: Outcomes for the internally recorded hypotheses. The protocol has no independently verified preregistration timestamp. Outstanding measurements are listed with the results.

	Hypothesis	Observed outcome
H1	storage $\leq P_A^u \times$ fact-inflated at equal accuracy	Observed candidate-pool ratio: leaf-Ω uses 48.9% of full-pool units at equal 97.45% served accuracy; physical storage not measured
H2	f_c-scheduled serves cheaper than unrestricted certified at equal served accuracy	-43% tokens under candidate restriction; scheduler-specific crossover remains unmeasured
H3	certified coverage $\geq$ L-B lower bound	Not empirically instantiated: logged outcome rates differ from the step-level channel rates; independence remains unverified
H4	erasure curves match $(1-\varepsilon)^\ell$ within Wilson CIs	Supported by Monte Carlo simulation
H5	zero reasoning shortcuts by construction	Structural admission replay rejects all 120 injected target writes; semantic leaf errors remain possible
H6	degradation rate $\leq$ C1 floor; no silent uncertified answers	Explicit refusal/status recorded; ContractNLI neutral and fallback outputs are uncertified
H7	kernel store smaller than retrieval index by $1/P_A^u$	Corpus redundancy measured; matched index-byte comparison deferred
H8	substrate parity at lower memory on M1–M2	Equal observed served accuracy with a smaller candidate pool; kernel-store byte accounting remains separate
H9	in-fragment accuracy $\geq$ pure/RAG baselines with zero silent answers (L1)	Supported (Table 3)
H10	every certified answer carries a derivation $\leq \delta + d_{\mathrm{rec}}$	Formal bounded-witness contract; complete per-query certificate archival was not instrumented in every adapter
P-M1	statutory layer stays at 0.60% in mixed corpus	**held**
P-M2	$\geq$ 10-point logical margin corpus-wide	**falsified as stated** (2.2 points); explained by the composition law (Corollary 2), which predicts the measured value within half a point
P-M3	instance/weakening dominates the redundant mass	**held** (91.7%)

F.1 LD-MH: Replication on the Real mathlib4 DAG

We applied the M-HARD generator to the mathlib4 dependency DAG extracted by the LeanDojo tracer. The generator uses the same depth strata 2–6, near-miss negatives, and twin-extended closure procedure for gold labels. The neural baselines remain near chance, with different errors on positive and negative queries. The certified channel answers nearly all queries with high accuracy. Table 9 reports the results on this second proof library.

F.2 External Validity on LeanDojo Benchmark 4

We also evaluate on *LeanDojo Benchmark 4* Yang et al. (2023), the official NeurIPS 2023 benchmark extracted from mathlib4 (commit `29dcec07`). We measure the four applicability criteria, premise selection, and multi-hop closure on this independently constructed corpus.

Four criteria on the real corpus. C1, C2, and C4 hold directly (120,509 theorems, 180,907 defined premises; 63.3% of annotated theorems multi-hop; depth long-tailed p_{50}=2, p_{90}=9, p_{99}=27). C3 remains unresolved because each traced theorem records only one derivation. The observed 36.5% premise usage measures which premises appear in those traces. Establishing redundancy would require testing whether omitted premises can be derived from the retained set. The AFP and regulatory audits perform such derivability checks.

LD-PS: premise selection. On 300 theorems sampled from the official `random/test` split (gold = premises of the traced proof; candidate pools of 15), the reference grounder scores: pure recall-from-memory 1.3% exact; BM25-pool RAG 11.3% exact (precision 85.6%); CPUNeSy (kernel-restricted pool, whitelist certificate, 2-vote abstention) 11.2% exact *on* 83% *coverage* with 98.9% *precision* (Table 8). On the two weaker grounders, CPUNeSy raises answered exact-set accuracy (DeepSeek 17.0% vs. RAG 8.7%; Qwen 19.9% vs. 12.0%), at 39.3/57.0% coverage. Full-pool exact-set accuracy is 6.7/11.3%, with gains concentrated in the answered subset. The reference run likewise has 9.3% full-pool exact-set accuracy versus BM25's 11.3%. Gold premises come from one traced proof among potentially many valid proofs; this limits the interpretation of exact-set matching. The models also select few premises, averaging 0.46–0.51 against a gold mean of 2.76. This pattern indicates frequent omission of needed premises.

LD-MH: closure judgment on the real DAG. The Seed 2.0 Lite cross-vendor run uses the same 200 depth-stratified tasks (99 pos /101 neg), hard distractors, and deterministic DAG-closure scorer as the synthetic protocol. Pure CoT obtains 50.0% and RAG 51.0%; one-vote grounding reaches 94.0%, while the two-vote certified channel obtains 93.9% on its 98.5% served subset and 92.5% full-pool accuracy (Table 9). Thus the certificate abstains on three tasks and preserves a 41.5-point full-pool advantage over the better neural control. This run tests the same closure task with a different grounding model from the primary K3 configuration.

LD-MH measures derivability over recorded dependency graphs. At 200/300 tasks, the Wilson half-widths are ±3–4 points. The measured execution advantage transfers to a second proof graph under the same task generator. End-to-end Lean proof search would require a separate evaluation.

Table 8: LD-PS ($n = 300$): full-pool and answered exact-set accuracy, premise precision/recall, and coverage. Abstentions count as errors in the full-pool column. These separate selection quality from the subset selected for answering.

Strategy	model	exact (all)	exact (served)	precision	recall	coverage
pure (recall from memory)	reference	1.3%	1.3%	70.0%	3.4%	100%
rag (BM25, full corpus)	reference	11.3%	11.3%	85.6%	14.4%	100%
CPUNeSy (kernel pool + cert. + 2-vote)	reference	9.3%	11.2%	**98.9%**	13.5%	83.0%
pure (recall from memory)	DeepSeek	2.0%	2.0%	9.0%	14.8%	100%
rag	DeepSeek	8.7%	8.7%	45.1%	22.0%	100%
CPUNeSy	DeepSeek	6.7%	**17.0%**	68.0%	31.4%	39.3%
pure	Qwen2.5-72B	1.3%	1.3%	7.1%	8.9%	100%
rag	Qwen2.5-72B	12.0%	12.0%	48.6%	34.0%	100%
CPUNeSy	Qwen2.5-72B	11.3%	**19.9%**	59.8%	43.8%	57.0%

Models select 0.46–0.51 premises on average, compared with a gold mean of 2.76.

Learned-retriever control (ReProver). We evaluate LD-PS candidate pools using the official pre-trained ReProver retriever (byt5-small encoder, trained by its authors on this benchmark's train split Yang et al. (2023)), run locally with no task-specific training. We use two retrieval protocols. *(i) Native tactic-state protocol*: over 2,641 traced test tactic states against the full 180,907-premise corpus, mean-pooled encoder embeddings and cosine ranking give per-premise recall R@1 = 8.3%, R@5 = 19.3%, R@10 = 26.0%, R@100 = 48.1%, and full gold-set coverage FC@10 = 18.0%. *(ii) LD-PS pool protocol*: a ReProver top-15 pool covers all gold premises for 9.0% of the 300 an-

chor tasks when queried by the theorem statement (the same input the LLM arms saw; recall@15 $= 18.5\%$), and 17.7% when queried by the first tactic state, which provides information unavailable to the LLM arms (recall@15 $= 31.5\%$). The BM25 and kernel-restricted pools both stand at 76.0% full coverage and 89.3% recall@15. BM25 and the kernel-restricted pool have higher coverage than ReProver in this candidate-pool evaluation. Their equal coverage makes BM25 the stronger retrieval control here. A storage comparison would also require measuring matched index sizes. ReProver normally ranks premises throughout proof search, whereas our tactic-state protocol evaluates only the first state. Larger or fine-tuned retrievers and complete proof-search runs remain to be tested.

Table 9: LD-MH (real mathlib4 DAG mirror of M-HARD, n=200). Seed 2.0 Lite is a completed cross-vendor control. Cells report full-pool accuracy except the coverage column; the Seed two-vote certified channel is 93.9% accurate on its served subset.

	pure	rag	1-vote	CPUNeSy	cov.
M-HARD (synthetic AFP, K3)	53.5	55.0	**96.5**	95.5	98.0
LD-MH (real mathlib4, Seed 2.0 Lite)	50.0	51.0	**94.0**	92.5	98.5
LD-MH (real mathlib4, DeepSeek)	51.5	50.0	**83.5**	79.0	89.5
LD-MH (real mathlib4, Qwen2.5-72B)	52.0	52.5	**77.5**	71.5	91.5

The 1-vote column is the deterministic-grounding ablation; CPUNeSy is the two-vote policy. High-budget M-HARD CoT and RAG score 60.5% $(121/200)$ and 59.5% $(119/200)$, respectively, so the margin over the stronger high-budget control is 35.0 points. The paired counts appear in Table 11. DeepSeek one-vote and coverage cells use the same frozen run (Appendix G).

F.3 MedCalc-Bench Verified: Protocol Notes

The pure baseline uses the official zero-shot CoT protocol. The open-book baseline adds calculator formulas to the context. CPUNeSy uses the model named in each column to extract entities, executes the verified reference implementations, and abstains on extraction disagreement (one vote) or disagreement between interfaces (two votes). The fallback cascade routes abstentions to the same model's open-book answer. All 55 reference implementations reproduce 100% of gold answers from gold entities, so residual errors come from the extracted inputs. Table 2 reports the main accuracy table for Seed and DeepSeek, with native two-vote full-pool accuracies of 72.5% and 63.5%, respectively. For Seed 2.0 Lite, the one-vote ablation serves $969/1{,}100$ items at 89.37% accuracy (78.73% full-pool); the two-vote channel serves $858/1{,}100$ at 92.89% accuracy (72.45% full-pool, rounded to 72.5% in the table). The updated DeepSeek result is reported at full-pool level; its two-vote channel serves $733/1{,}100$ items, with 698 correct, for 95.23% answered accuracy, 66.64% coverage, and 63.45% full-pool accuracy (rounded to 63.5% in the table). The recorded result file retains item-level records across all 55 calculators for stratified analysis.

Supplementary GPT-5.2 run. A supplementary run used GPT-5.2 through an aggregator API. With no fallback, its two-vote channel reaches 91.16% answered accuracy at 72.0% coverage (65.64% full-pool). The Seed–DeepSeek comparison uses the model profiles in Table 12.

F.4 Fairness and Robustness Controls (Full)

The internal protocol records hypotheses H1–H10 and P-M1–P-M3 with falsification thresholds (Table 7). The task generators are machine-checked, and Appendix G describes protocol provenance.

A weaker grounding model retains a $\sim$36-point advantage on the compositional stratum against frontier baselines using $\leq 3\times$ budget. Regenerating each suite with 3 seeds preserves the reported performance gaps. The gaps also persist under query paraphrases and across Jaccard-ranked top-1, top-5, and random distractor pools.

We report Wilson 95% intervals for rates and exact McNemar tests for paired main comparisons. The uncertified-solver comparison gives a two-sided p=0.34 (b=3, c=7).

F.5 Observed Error Counts and Channel-Model Scope

The recorded error counts characterize observed outcomes at several levels. For L1-HARD and LD-MH, respectively, the logs report $20/1{,}283$ and $34/1{,}853$ grounding-item failures; $0/457$ and

$15/301$ wrong verdicts conditional on clean grounding; $20/477$ and $27/315$ incorrect accepted traces; and $9/235$ and $11/192$ wrong agreed outputs in the broad-interface two-vote runs. These fractions have different denominators and describe different events. The 20 L1-HARD trace errors include assertions of the queried conclusion as a premise. Separately, the restricted leaf-interface policy yields no observed wrong answers among 233 served cases; changing interfaces also changes the proposals supplied to voting (Appendix G). The reported LD-MH errors conditional on clean grounding concentrate on negatives with intermediate-node routes around the broken leaf in the recorded dependency graph.

These outcome rates do not directly estimate the parameters of Proposition 2. In Definition 6, μ bounds loss or corruption of an intended derivation step, and η bounds rejection of a *valid* step. An incorrect accepted trace is a different event from a valid step being rejected. Likewise, an aggregate grounding-item failure fraction is not by itself a uniform upper bound on each exposed premise's formalization failure. We therefore report the counts as descriptive error diagnostics and do not substitute them into the coverage bound. Empirical instantiation would require annotations of the corresponding step-level events, appropriate probability bounds, and assessment of the stated independence assumptions. The proposition remains a conditional guarantee.

G Admission Audit and Reproducible Offline Analysis

G.1 Source checking and fixed-trace gate effect

ContractNLI uses a learned rechecker on the source document and each contributing fact. A fact must receive `holds`; otherwise the vote abstains. Conflicting closure results also abstain. Two agreeing entailment/contradiction votes set the certificate flag; two neutral votes produce the separate `neutral_uncertified` status. The main native-policy row includes this explicitly uncertified neutral status, whereas strict certificate-only statistics require the flag.

The gate ablation compares raw two-vote agreement with the stored output after rechecking, using the same grounding traces. Seed changes (C, W, A) from $(858, 128, 51)$ to $(844, 120, 73)$; DeepSeek changes from $(1533, 252, 306)$ to $(1482, 189, 420)$. Rechecking withholds answers while leaving all other outputs unchanged. The counts measure which correct and wrong answers are withheld. Estimating premise-level false acceptance and rejection would require independent source-support annotations, which these records lack.

MedCalc parses calculator-specific fields, normalizes units and values, requires agreement, and calls the deterministic calculator. The recorded one-vote and native two-vote policies give $(C, W, A) = (866, 103, 131) \rightarrow (797, 61, 242)$ for Seed and $(824, 88, 188) \rightarrow (698, 35, 367)$ for DeepSeek. This comparison includes the agreement requirement and is distinct from the ContractNLI source-recheck ablation. The agreed, schema-valid attributes can still misrepresent the clinical note. The risk estimates include the resulting wrong answers.

G.2 L1-HARD replay and structural admission

The historical leaf interface specifies the allowed vocabulary in the prompt. Its executor accepts any parsed predicate. We therefore provide a separate explicit admission replay that rejects predicates outside the rule-derived leaf vocabulary. All parsed proposals in the recorded restricted runs use leaf predicates. The offline structural stress test uses generator facts restricted to allowed leaves as clean controls, then adds the target atom to each negative item. The barrier rejects $120/120$ injected target writes in this constructed test. The raw solver accepts every injected target by construction. Generator facts supply the clean replay controls and scoring labels. The historical model-serving runs obtain their proposed facts from the model.

The broad two-vote joint outcomes are 226 both correct, 9 both wrong, 2 correct/wrong, and 3 pairs with one parsing failure. The leaf two-vote outcomes are 233 both correct, 2 correct/wrong, 2 wrong/correct, and 3 pairs with one parsing failure. Thus the historical $20/477$ single-trace errors belong to the broad interface, while the $0/233$ served-error observation belongs to the leaf interface. Comparing these runs changes both the vocabulary and the proposals supplied to the voting rule. Against exact generator source-fact strings, the $1{,}804$ distinct-per-trace leaf proposals contain $1{,}732$

matches and 72 mismatches. These are exact string matches. Assessing whether the mismatches are semantically supported would require human review of the source.

G.3 Statistics, latency, and calibration

All policies share their task denominator; parse failures count as abstentions. The L1 accuracy-difference interval uses 20,000 paired item bootstrap resamples with seed 20260926. The $0/233$ upper error bound is $1 - 0.05^{1/233}$ under an item-binomial sampling model. The comparisons are exploratory and report counts and confidence intervals. Table 11 gives Holm-adjusted tests for the six paired comparisons. Two traces are insufficient to estimate the theorem's latent β_{ans} and σ_c separately.

A prospective deployment protocol would freeze a disjoint calibration set with source annotations, label both-wrong, discordant, and invalid traces, measure accepted-answer risk and coverage, choose c_W, c_A, c_T before testing, then freeze the write vocabulary and serving rule on a separate test set. Our experiments report the recorded joint outcomes for retrospective analysis. Testing the proposed deployment rule on held-out data remains future work. We calculate latency by summing logged call durations. The logs lack complete measurements of deployment formalization, expert review, concurrency, and monetary cost.

G.4 Protocol provenance

The artifact contains a dated internal mixed-corpus protocol and a record of its hypotheses. There is no independent timestamp establishing preregistration. Scripts and input hashes make the present analyses reproducible. Table 7 reports the outcome of each hypothesis and the measurements still needed.

G.5 Supplementary gate effect and cost analysis

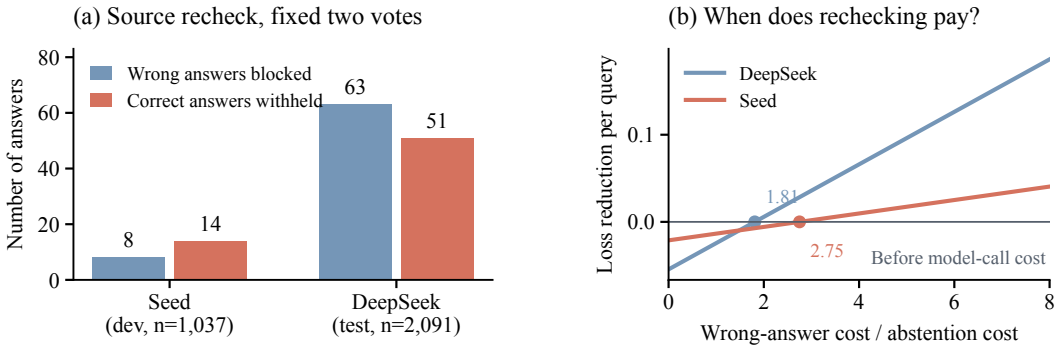


Figure 7: Additional fixed-trace source-recheck analysis on ContractNLI. The left panel counts correct and wrong answers withheld. The right panel uses these counts to calculate loss reduction at different relative error costs.

For policies on the same N items, let c_W, c_A, c_T denote the respective costs of a wrong answer, refusal, and token. The loss difference is

$$\Delta L = (\Delta W c_W + \Delta A c_A + \Delta T c_T)/N. \tag{1}$$

Before compute cost, ContractNLI source rechecking improves this loss when $c_W/c_A > 114/63 = 1.81$ for DeepSeek and $> 22/8 = 2.75$ for Seed. Restricted L1-HARD two-call serving is preferred when $2c_W > 6c_A + 145{,}422c_T$ across 240 items. The paired accuracy-difference interval is $[-4.17, 0.83]$ percentage points (two-call minus single-call), which does not establish equivalence. Median sums of logged call durations are 6.97 versus 13.88 seconds. These measurements assume sequential calls.

G.6 SELECTIVE OPERATING POINTS AND PAIRED TESTS

Table 10 reports discrete serving policies, with wrong answers divided by served answers for risk and served answers divided by the common pool for coverage. Each row is a discrete policy. Estimating a risk–coverage curve or its area would require a confidence-threshold sweep. Wilson intervals use the item-level sampling model.

Table 10: Matched error–coverage accounting from frozen records. C/W/A denote correct/wrong/abstained counts. Risk and coverage are percentages; brackets are 95% Wilson intervals for answered risk.

Data / model	Policy	C	W	A	Coverage	Risk [95% CI]
ContractNLI Seed	Agreement only	858	128	51	95.1	13.0 [11.0, 15.2]
	Native policy	844	120	73	93.0	12.4 [10.5, 14.7]
	Certificate flag only	513	75	449	56.7	12.8 [10.3, 15.7]
	+ fallback	880	157	0	100.0	15.1 [13.1, 17.4]
ContractNLI DeepSeek	Agreement only	1533	252	306	85.4	14.1 [12.6, 15.8]
	Native policy	1482	189	420	79.9	11.3 [9.9, 12.9]
	Certificate flag only	872	119	1100	47.4	12.0 [10.1, 14.2]
	+ fallback	1742	349	0	100.0	16.7 [15.2, 18.3]
MedCalc Seed	One vote	866	103	131	88.1	10.6 [8.8, 12.7]
	Native policy	797	61	242	78.0	7.1 [5.6, 9.0]
	+ fallback	991	109	0	100.0	9.9 [8.3, 11.8]
MedCalc DeepSeek	One vote	824	88	188	82.9	9.6 [7.9, 11.7]
	Native policy	698	35	367	66.6	4.8 [3.5, 6.6]
	+ fallback	994	106	0	100.0	9.6 [8.0, 11.5]
LD-MH Seed	One vote	188	12	0	100.0	6.0 [3.5, 10.2]
	Native policy	185	12	3	98.5	6.1 [3.5, 10.3]
	+ fallback	185	15	0	100.0	7.5 [4.6, 12.0]
LD-MH DeepSeek	One vote	167	33	0	100.0	16.5 [12.0, 22.3]
	Native policy	158	21	21	89.5	11.7 [7.8, 17.3]
	+ fallback	167	33	0	100.0	16.5 [12.0, 22.3]

Table 11: Paired full-pool accuracy tests. b counts CPUNeSy-only correct items; c counts baseline-only correct items. Abstentions count as incorrect. Holm correction is applied jointly to these six exploratory comparisons.

Comparison	b	c	Exact p	Holm p
M-HARD vs pure	92	8	3.21×10^{-19}	1.60×10^{-18}
M-HARD vs rag	87	6	1.65×10^{-19}	9.90×10^{-19}
M-HARD vs pure_hi	78	8	1.52×10^{-15}	4.57×10^{-15}
M-HARD vs rag_hi	77	5	1.20×10^{-17}	4.82×10^{-17}
L1-HARD vs pure	36	6	2.83×10^{-6}	5.66×10^{-6}
L1-HARD vs rag	28	6	1.95×10^{-4}	1.95×10^{-4}

G.7 REPRODUCIBLE ARTIFACT AND RECORDED MODEL IDENTITIES

The accompanying offline audit package contains the frozen input records, source-file SHA-256 hashes, scripts, and generated counts. Its README gives commands that require no API credentials or network calls. Table 12 lists the available model identifiers. The caches lack consistent records of sampling dates and immutable model revisions. Two-vote majority with abstention on ties is identical to the binary agreement rule on the same valid traces. The fixed-trace ablation therefore measures the additional effect of source rechecking after this voting rule. The archive preserves all policy counts, including incorrect accepted answers and explicitly uncertified neutral/fallback statuses.

G.8 PROCESS-FIGURE PROVENANCE

Figures 3–5 use K3 for L1-HARD and LD-MH, and Seed for ContractNLI and MedCalc. The LD-MH geometry panel shows the $181/19$ split from K3, while the public accuracy table reports the $185/15$ split from Seed. The plotted features summarize grounding, admission, closure, and output logs. Principal component analysis (PCA) and adjusted Rand index (ARI) describe where

Table 12: Recorded model names and their scope. Primary public columns use the profiles identified here; historical adapter runs are a separate campaign.

Paper label	Recorded identifier	Scope / provider record
K3 / reference	`k3-agent`	Controlled L1-HARD and M-HARD; model profile configured by experiment environment
K2.6 / weaker same-family	`k2d6-agent`	Same-family control; model profile configured by experiment environment
DS (L1 controls)	`deepseek-chat`	Cross-model L1 study; distinct from public-column Flash
QW	`Qwen/Qwen2.5-72B-Instruct`	Cross-model control accessed through a public API
GLM	GLM-5.2	Cross-model control; provider label in the experiment record
Seed (public columns)	Seed 2.0 Lite (profile identifier redacted)	Recorded profile Seed 2.0 Lite
DeepSeek (public columns)	`deepseek-flash`	Official DeepSeek API profile
GPT-5.2	GPT-5.2 aggregator run	Supplementary MedCalc only

the recorded protocols diverge. The fixed-trace comparison in Section 4.2 measures the effect of source rechecking.

G.9 Historical Task-Adapter Results

Table 13 preserves the historical task-adapter summaries. These runs use other splits and model profiles than the primary matched comparisons. Their per-row split identifiers, model profiles, and sample counts are not specified in the available summaries, so the archived column labels do not establish matching to Table 2. We do not use these scores to rank the primary CPUNeSy policies against the original systems. Near-zero scores also require a failure breakdown: the summaries do not separate parsing failures, solver failures, and wrong executable outputs. The aggregate scores alone cannot determine their cause.

Table 13: Historical task-adapter full-pool accuracy (%). Column names reproduce archive labels, not matched model/split identities. Exact per-row denominators are unspecified in these summaries; these results are descriptive and excluded from primary comparative claims.

	ContractNLI		MedCalc		LD-MH	
Adapter	Seed	DeepSeek	Seed	DeepSeek	Seed	DeepSeek
Logic-LM (Pan et al., 2023)	0.0	0.0	75.5	77.5	50.5	46.5
DetermLR (Sun et al., 2024)	48.4	70.0	67.5	61.5	50.5	49.5
LINC (Olausson et al., 2023)	16.0	0.2	66.5	44.7	49.5	51.0
SymbCoT (Xu et al., 2024)	56.8	16.5	66.5	75.1	49.5	49.5
LINA (Li et al., 2024a)	48.5	10.8	67.3	69.3	49.5	51.0
HBLR (Li et al., 2025)	54.7	10.4	66.3	62.8	49.0	50.5
MenTaL (Li et al., 2026)	1.8	0.0	75.0	76.2	47.5	50.0

H Supplementary Visualizations

Figures 8, 11, and 13 illustrate the theoretical models with schematic curves. The other panels report stress-test results or Monte Carlo erasure trials.

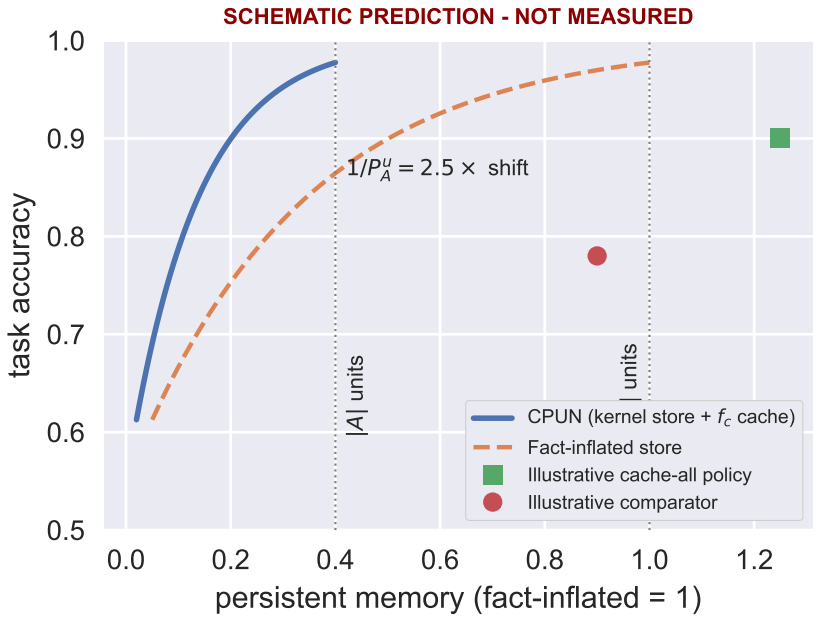


Figure 8: Illustrative memory–accuracy prediction under the kernel-storage model. The curves and reference points are schematic. The legacy legend label CPUN denotes CPUNeSy.

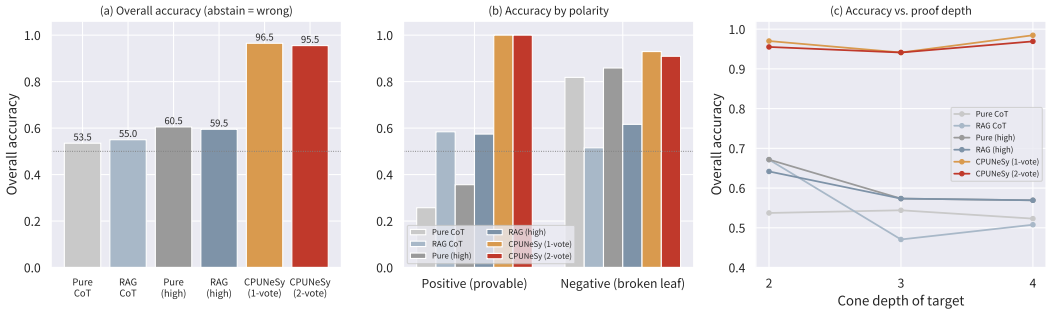


Figure 9: M-HARD visual summary. CPUNeSy separates from pure and retrieval baselines in overall accuracy, removes the polarity asymmetry on positives versus near-miss negatives, and stays stable as proof depth increases.

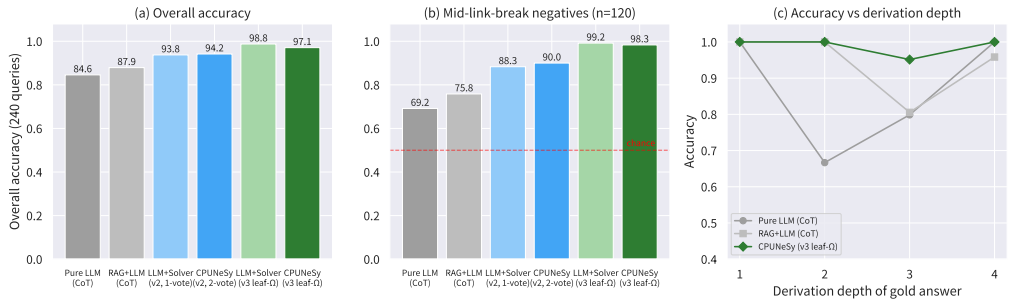


Figure 10: L1-HARD visual summary. CPUNeSy's accuracy gain is concentrated on mid-link-break negatives, where factual access alone is insufficient. Section 4.2 measures each component's contribution through ablation.

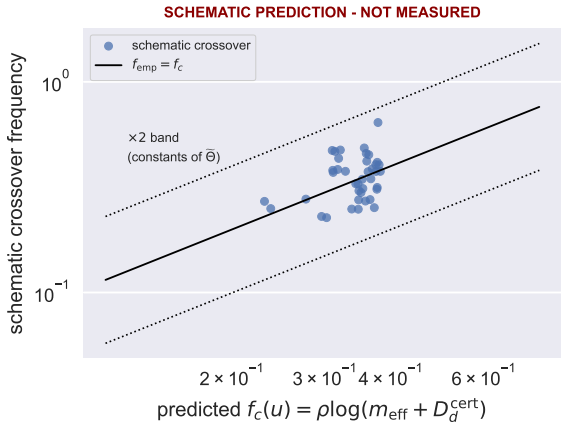


Figure 11: Illustrative crossover prediction under the stationary store–compute model. The schematic points and constant-factor band illustrate the predicted relationship between the caching threshold and crossover frequency.

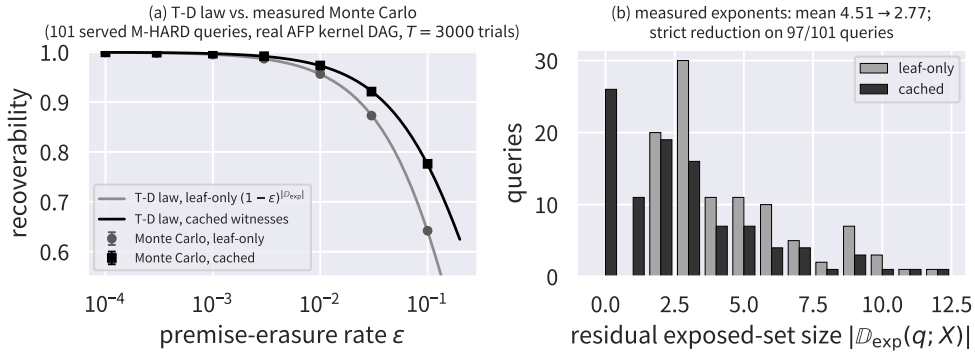


Figure 12: Monte Carlo premise-erasure recovery on 101 served M-HARD queries and their AFP witness DAGs (3,000 trials). The left panel compares recovery with the residual-leaf law. The right reports exposed-set sizes before and after caching. T-D denotes Corollary 1. The simulation assumes storage loss with a surviving cache and unchanged source support.

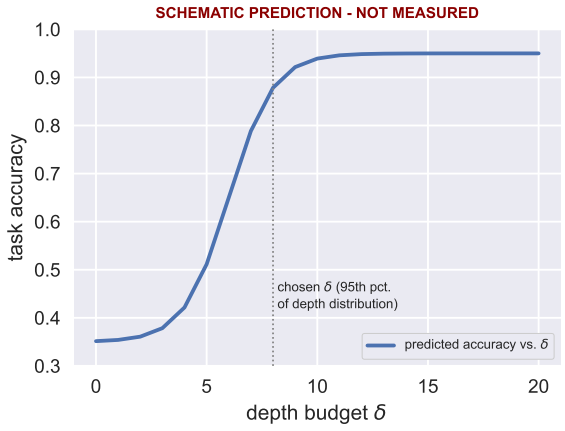


Figure 13: Illustrative depth-budget sensitivity prediction. The schematic curve illustrates saturation near a budget chosen from the depth distribution.

I CROSS-MODEL EVALUATION

I.1 CROSS-MODEL ERROR REGIMES ON L1-HARD

Table 14 compares five models from four model families using identical prompts, solver, and parsers. The reference, weaker same-family, and GLM runs retain high answered accuracy. DS and QW have more shared errors. GLM produces 2 pairs that agree on a wrong answer, and the agreement rule filters 15 disagreements. These joint outcomes provide the retrospective error classifications used in Section 4.3. Full-pool accuracy counts abstentions as errors.

Table 14: Cross-model replication on L1-HARD (240 queries; identical prompts, solver, and parsers on all models). Reference-grounder numbers are shown in Table 3; weaker same-family, DS, and QW rows carry Wilson 95% intervals on overall accuracy. *neg* = accuracy on the 120 mid-link-break negatives. DS (DeepSeek-chat) and QW (Qwen2.5-72B-Instruct) are different vendors/architecture lineages.

Strategy	model	overall [CI]	neg	tok/q
Pure CoT	K3	84.6	69.2	611
	K2.6	82.9 [77.6, 87.2]	74.0	1,187
	DS	73.3 [67.4, 78.5]	46.7	474
	QW	90.0 [85.6, 93.2]	80.8	459
	GLM	92.9 [89.0, 95.5]	86.7	1,627
RAG+CoT	K3	87.9	75.8	759
	K2.6	88.3 [83.7, 91.8]	80.0	1,196
	DS	84.6 [79.5, 88.6]	71.7	450
	QW	87.1 [82.2, 90.7]	75.0	640
	GLM	92.5 [88.5, 95.2]	85.0	1,525
Solver 1-vote	K3	98.8	99.2	606
	K2.6	96.3 [93.0, 98.0]	99.2	795
	DS	83.3 [78.1, 87.5]	82.5	328
	QW	84.6 [79.5, 88.6]	95.8	351
	GLM	96.2 [93.0, 98.0]	100	1,118
CPUNeSy 2-vote	K3	97.1	98.3	1,212
	K2.6	**94.2** [90.4, 96.5]	**99.2**	1,551
	DS	76.7 [70.9, 81.6]	77.5	661
	QW	82.1 [76.7, 86.4]	89.2	709
	GLM	92.9 [89.0, 95.5]	100	2,406

served accuracy (coverage): reference 100% (97.1%); weaker same-family **99.6**% [97.6, 99.9] (94.6%), disjoint from pure 86.5% [81.4, 90.3] and RAG 90.2% [85.7, 93.4]; GLM **99.1**% [96.8, 99.8] (93.8%), with a confidence interval disjoint from its own pure 92.9% and RAG 92.5% results. GLM has 2 agree-wrong pairs and filters 15/15 disagreements; DS 83.6% [78.2, 87.9] (91.7%) and QW 88.0% [83.0, 91.6] (93.3%), on par with their neural baselines, consistent with the shared-error boundary of Lemma 1. All grounded rows use the leaf-Ω interface. DS = DeepSeek-chat; QW = Qwen2.5-72B-Instruct; GLM = GLM-5.2.