GENERALIZED GROKERS: THE VERIFIER-RICH SUBSTRATE IS AUTONOMOUSLY GROKABLE

Anonymous authors
Paper under double-blind review

ABSTRACT

The verifier-rich structure that makes trust a computable property of LLM inference can be discovered and maintained autonomously from the corpus itself. An inference run over a trajectory has three reusable artifacts: the *strategy* that orders its steps, the *function* computed at each step, and the *context* each step conditions on. One amortization principle governs all three: keep the reusable artifact off the per-query path and learn it incrementally from actual data, so a model invocation becomes a cheap deterministic read. Probabilistic Language Tries amortize the strategy, LAWS amortizes the function as a certified expert, and groking amortizes the context as an incrementally maintained typed comprehension index.

We formalize groking as such an index, give an amortized-comprehension bound stating when it beats per-query re-derivation, and show that its bottom-up order is the operational form of certified composition. An upward-closed uncertainty result confines the index's model-quality residual to an explicit, shrinking flagged set; and, made conservative, the index is a sound over-approximation at every depth for a model of any accuracy—by order-theoretic monotonicity rather than any smoothness assumption—so model quality sets its precision, never its soundness. We show the domains admitting profitable groking are exactly the verifier-rich ones: code, games, formal mathematics, hyperlinked corpora, schemas, and data pipelines.

1 INTRODUCTION

A forward pass is stateless. The same model, however well aligned, computes the same function for every query and carries nothing from one invocation to the next, so whatever makes one deployment more capable or more reliable than another running the same model is not in the model. It is in what the system remembers, how that memory is organized, and how the relevant part is brought to bear.

A deployed LLM system pays two recurring costs that have nothing to do with the difficulty of any single query. It re-runs computations the model has effectively performed before, and it re-derives the structure of context it has already comprehended. The companion paper addresses the first: it learns cheap certified experts from a workload, so recurring computation is read from a cache rather than re-executed by the model, and it makes the resulting guarantees a computable property of the system. That paper takes the substrate—the certifiers, the cheap experts, and the explicit structure they check against—as given. This paper is about where that substrate comes from.

The claim is that the substrate need not be built by hand. An inference is $f(\text{context})$, and across a trajectory three artifacts recur and are reusable: the *strategy* (what sequence of steps to take), the *function* (what computation to run at a step), and the *context* (what each step conditions on). One principle handles all three: *keep the reusable artifact off the per-query path and learn it incrementally from actual data*.

Contributions.

1. The amortization principle and the three axes—strategy, function, context—with the signpost as the shared symbolic anchor (Section 2).

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.

2. Groking formalized as an incrementally maintained typed comprehension index, with an amortized-comprehension bound (Proposition 1) stating when it beats per-query re-derivation (Section 3).
3. The identification of bottom-up comprehension with certified composition (Section 4).
4. The soundness result: a conservative index is a sound over-approximation at every depth for a model of any accuracy (Theorem 1).
5. A grokability criterion: the domains admitting profitable groking are exactly the verifier-rich ones (Section 8).

2 The Amortization Principle

Three axes run through every sustained inference trajectory, each reusable and each currently paid per query:

Strategy. The next-step decision recurs: "given what I know, what do I ask next?" Probabilistic Language Tries amortize it by learning the workload's trajectory distribution, making the next step a single trie lookup rather than a chain-of-thought pass.

Function. The same function is invoked repeatedly—a retrieval, a transformation, a reasoning step. LAWS amortizes it by learning a cheap certified expert from traces, so the function is executed by a fast specialization verified correct rather than by the full model.

Context. Each step conditions on accumulated context the model has effectively already comprehended. Groking amortizes it by maintaining the comprehended structure as a typed graph, so context is read rather than re-derived.

All three share the same lever: move the model off the per-query path. Each trades write-time cost for read-time savings, and each is profitable in the same regime—read-mostly, derive-expensive workloads where one write is amortized over many reads.

3 The Comprehension Index

Definition 1 (Comprehension index)**.** *For a corpus D with reference structure—items u carrying typed edges $u \to v$—a* comprehension index $\Gamma(D)$ *is a persistent typed graph whose nodes hold comprehended attributes (summaries, contracts, extracted relations) computed bottom-up over the reference DAG, maintained incrementally so that a write touching items U recomputes only U and their transitive dependents. The context for a query q is a read* $\mathrm{Ctx}(q) = \mathrm{Read}(\Gamma(D), q)$*—a traversal of $\Gamma(D)$ rather than a fresh derivation from D.*

Maintaining $\Gamma(D)$ is *groking*. Concretely, a parse pass extracts items and their typed edges (call edges, dependency edges, lineage edges); an analysis pass comprehends each node bottom-up, leaves before roots, so that by the time a node is reached its dependencies are already understood; and the result is materialized as a typed graph that subsequent queries traverse in time proportional to the relevant subgraph, with no model call.

Proposition 1 (Amortized comprehension)**.** *Over a window of Q queries and W writes, let κ_{der} be the cost to derive the query-relevant structure from the raw corpus, κ_{rd} the cost to read it from $\Gamma(D)$, and κ_{mnt} the amortized cost to maintain $\Gamma(D)$ per write ($\kappa_{\mathrm{rd}} < \kappa_{\mathrm{der}}$). Per-query re-derivation costs $Q\kappa_{\mathrm{der}}$; groking costs $W\kappa_{\mathrm{mnt}} + Q\kappa_{\mathrm{rd}}$. Groking is cheaper iff*

$$\frac{Q}{W} > \frac{\kappa_{\mathrm{mnt}}}{\kappa_{\mathrm{der}} - \kappa_{\mathrm{rd}}}.$$

Proof. $W\kappa_{\mathrm{mnt}} + Q\kappa_{\mathrm{rd}} < Q\kappa_{\mathrm{der}} \iff W\kappa_{\mathrm{mnt}} < Q(\kappa_{\mathrm{der}} - \kappa_{\mathrm{rd}})$; divide by $W(\kappa_{\mathrm{der}} - \kappa_{\mathrm{rd}}) > 0$. □

This is the incremental-view-maintenance law: pay maintenance on write to save derivation on read, profitable in the read-mostly, derive-expensive regime.

The lever is the per-query model call. Write ℓ for the cost of one model invocation over the query-relevant context and $d \ll \ell$ for a deterministic graph operation. Re-derivation comprehends that context at query time, so $\kappa_{\mathrm{der}} \geq \ell$: the model sits on the per-query path. Reading from $\Gamma(D)$ is a traversal, $\kappa_{\mathrm{rd}} = \Theta(ds)$, with *no model call*. The model invocations move to write time.

Corollary 1 (Per-query model invocations)**.** *Let $q \in [0, 1]$ be the fraction of queries not covered by the materialized schema. The expected model invocations per query is 1 under re-derivation and q under groking. Once the index is warm, the model is touched a factor $\approx 1/q$ less per query.*

These are not only asymptotic concerns. The re-derivation regime is the $153{:}1$ input-to-output blowup measured across eight frontier agents on real coding tasks, where every loop re-ingests the accumulated context. Moving the model off the per-query path is the structural cost reduction.

Corollary 2 (Cross-organization amortization)**.** *A capability generated and verified once at cost G and reused across M runs contributes amortized generation cost G/M per run—Proposition 1 at the capability level. Moreover an interface novel for the first consumer is a read for every later one, so the aggregate novelty fraction of a population sharing a verified library is non-increasing in the library's coverage. Broadening shared coverage raises the achievable reduction for the whole population.*

4 Bottom-Up Comprehension Is the Composition Order

The bottom-up order is not a choice among equals. A node's contract is a function of its dependencies' contracts; the DAG's leaves have no dependencies and can be comprehended first; their parents become comprehensible once the leaves are done; and so on to the roots. This is *certified composition*: each layer is verified against premises that are themselves already verified, and the verification propagates.

Concretely: a certifier χ at node v checks that the contract $\hat{c}(v)$ proposed by the model is *sound*—that it over-approximates v's true behavior—given the contracts of v's dependencies. If the dependencies are not yet comprehended, the check cannot run; bottom-up order ensures they always are.

5 Soundness at Every Depth

The structural skeleton of $\Gamma(D)$ is sound; the comprehended attributes are model-quality. That gap is not unbounded. Let $U \subseteq V$ be the *flagged set*: nodes the analyzer could not resolve with confidence—dynamic dispatch, reflection, `eval`, behavior set by runtime configuration—each recorded with an explicit concept-to-investigate.

Proposition 2 (Upward-closed uncertainty)**.** *If a node's comprehension depends on a child through a property the child's summary does not determine, the node inherits a flag. Hence U is closed upward along dependency edges, and a node outside U has attributes determined entirely by sound, summary-determined properties of its children. U is therefore a sound over-approximation of the nodes whose attributes may be wrong.*

Comprehension is thus sound on $V \setminus U$, and the model residual is confined to U: explicit, finite, queryable, and shrinking as concepts are resolved.

Conservative certification. The model is not assumed accurate. Instead, the index is made *conservative*: a certifier accepts a proposed contract only when the composition rule, applied to the consumed children's contracts, yields something weaker than or equal to the proposal. A flagged child is read as $\top$, the weakest contract.

Let contracts form a lattice $(\mathcal{C}, \sqsubseteq)$ in which $c^* \sqsubseteq \hat{c}$ means $\hat{c}$ claims no more than the truth—a sound over-approximation—with top element $\top$ the least informative contract, sound for anything. Write F_v for the assume-guarantee composition yielding a node's contract from its children's, monotone in $\sqsubseteq$ (weaker children give a weaker parent, as in Hoare-style composition).

Theorem 1 (Soundness at every depth)**.** *Under conservative certification and monotone composition, every certified node satisfies $c^*(v) \sqsubseteq \hat{c}(v)$: its comprehended contract is a sound over-approximation of the truth, at every depth and for a model of any accuracy. Model quality changes only which nodes are flagged, never the soundness of those that are not.*

Proof. Induction in bottom-up order. Every consumed child contract over-approximates the truth: an accepted child by the inductive hypothesis, $c^*(u) \sqsubseteq \hat{c}(u)$; a flagged child because it is read as $\top \sqsupseteq c^*(u)$. The composition rule is sound: v satisfies $F_v(\{c^*(u)\})$, and since $c^*(v)$ is the strongest contract v satisfies, $c^*(v) \sqsubseteq F_v(\{c^*(u)\})$. By monotonicity $F_v(\{c^*(u)\}) \sqsubseteq F_v(\{\text{consumed}\})$, and conservative acceptance gives $F_v(\{\text{consumed}\}) \sqsubseteq \hat{c}(v)$. Chaining, $c^*(v) \sqsubseteq \hat{c}(v)$. □

Corollary 3 (Imprecision is additive, not amplifying)**.** *Measure informativeness by a monotone height h and let each F_v be non-expansive in h. Then the imprecision $h(\hat{c}(v)) - h(c^*(v))$ is at most the sum of local losses over v's descendants. Imprecision accumulates additively in the number of imprecise descendants and is otherwise independent of depth.*

Theorem 2 (Monotone refinement in certifier power)**.** *If $\chi' \succeq \chi$ (a stronger certifier) then $c^*(v) \sqsubseteq \hat{c}_{\chi'}(v) \sqsubseteq \hat{c}_{\chi}(v)$ at every node: strengthening the layer refines the sound index pointwise and never breaks soundness.*

Corollary 4 (Convergence to the verifier-poor core)**.** *Along a rising chain of certifiers, the indices converge to the strongest sound index the layer can reach. The residual flagged set is the verifier-poor core: the nodes no certifier in the chain discharges, fixed by what is checkable and not by the model.*

This is the lever the thesis rests on: the model's accuracy moves realizability against a fixed frontier and never soundness, while the certifier's power moves the frontier itself, monotonically, bottoming out at verifier-richness.

6 Amortization Across Agents

The amortization so far lives inside one agent: comprehend once, read many times. The same principle crosses *agents*. When two collaborate—two teams negotiating an API change, two services reconciling a schema—each turn that re-ships its context pays to re-transmit what the other could already name. Content-addressing removes that cost.

Definition 2 (Content-addressed citation)**.** *Agents first exchange the hashes of their contexts and transfer only the items the other lacks—the delta D. Thereafter an item is cited by its hash, at $O(1)$ cost, rather than by its body.*

Theorem 3 (Amortization across agents)**.** *The content-addressed cost of the collaboration is $|D| + \sum_t \nu_t$, against a naive $\sum_t |\text{cited}_t|$ that re-ships cited context each turn. It is independent of T (the number of turns) except through novelty: if per-turn novelty $\nu_t \to 0$, the total tends to $|D|$, a constant in the number of turns.*

Proposition 3 (Information floor)**.** *Any protocol under which each agent can reconstruct the cited context must transmit at least the content not already shared—the novelty. Content-addressing attains this up to an $O(1)$ hash per item, so $\sum_t \nu_t$ is irreducible and the protocol is optimal up to hashing overhead.*

Single-agent comprehension amortizes across queries; this amortizes across agents. Across three dimensions—queries within an agent, agents within a collaboration, and edits over time—the law is a single statement. The irreducible cost is the new information: the novelty rate on the query axis, the context delta on the agent axis, the change frontier on the edit axis. Everything else is a lookup.

7 Autonomous Strategy Grokking

The same principle applies to strategies. A strategy is a mapping from a state to a next step; groking a strategy is learning that mapping from traces so that the next step is a deterministic lookup rather than a chain-of-thought pass.

Definition 3 (Grokked strategy)**.** *A strategy $\sigma : S \to A$ is* grokked *when its mapping is materialized in a data structure that returns $\sigma(s)$ in time sublinear in $|S|$, learned incrementally from actual trajectories rather than given by hand.*

Topological refactoring. The worked instance is refactoring a codebase. The strategy is "what file to refactor next?" and the constraint is that a file must be refactored only after everything it depends

on. The grokked strategy is the topological sort of the dependency graph, learned once from static analysis and reused for every refactoring pass. No chain-of-thought reasoning is needed to decide the order; the order is read from the materialized DAG.

8 When a Domain Is Grokable

A domain admits profitable groking when it satisfies Proposition 1's threshold and permits Theorem 1's construction. Both conditions reduce to the same structural property: the domain must be *verifier-rich*—it must admit cheap, compositional certifiers that check proposals against premises.

Grokable domains.

- **Code.** Type-checker, linter, test harness. Each function has a signature; composition is call-graph traversal; the certifier checks type consistency, precondition satisfaction, side-effect bounds.
- **Formal mathematics.** Proof assistant. Each lemma has a statement; composition is proof dependency; the certifier checks logical validity.
- **Games.** Rules engine, self-play verifier. Each position has a value; composition is game-tree traversal; the certifier checks move legality and win conditions.
- **Hyperlinked corpora.** Link-graph consistency. Each document has anchors; composition is hyperlink traversal; the certifier checks link validity and citation consistency.
- **Schemas and data pipelines.** Constraint checkers, lineage validators. Each table has a schema; composition is dataflow traversal; the certifier checks constraint satisfaction.

Each has explicit typed structure and local certifiers that run in polynomial time.

Non-grokable domains. Open-ended prose without links or citations; unstructured image corpora; domains where the only check is another model call. Here the certifier is as expensive as derivation, the threshold is never crossed, and the soundness theorem has no local layer to exploit.

The conclusion is a characterization: the domains that admit autonomous groking are exactly the verifier-rich ones, and the index that results is exactly what certification needs.

9 Related Work

Incremental view maintenance. The database ancestor of groking maintains materialized views incrementally, paying on write to save on read. The contribution here is that the maintained artifact holds LLM-comprehended *semantic* structure—summaries and contracts, not only syntactic links—and that the soundness theorem (Theorem 1) makes the maintained structure certifiably sound.

Retrieval-augmented generation. RAG places context retrieval on the per-query path; groking moves comprehension to write time and retrieves from a pre-comprehended index. The distinction is whether the model is invoked at read time (RAG) or only at write time (groking).

Certified inference. LAWS and Generalized LAWS formalize the function axis, learning certified experts from traces. The context axis here completes the picture: the model manufactures the structure those certifiers check, autonomously and soundly.

Program analysis and comprehension. Static analyzers compute facts about code bottom-up; type systems compose contracts through assume-guarantee reasoning. Groking generalizes this to LLM-assisted comprehension, where the model proposes and the certifier checks, and the soundness theorem (Theorem 1) recovers the compositional guarantee.

Knowledge graphs and semantic indexing. Knowledge graphs store explicit typed relations; groking maintains those relations incrementally from a changing corpus, with the soundness guarantee that certified nodes over-approximate the truth.

10 CONCLUSION

Trust is a computable property of LLM inference, and the verifier-rich structure that makes it computable is autonomously grokable. The model in all of this is generic and stateless; what makes one deployment more capable than another running the same model is the organized memory it apprehends and consults, and the guarantees here are what let that memory accrue without accruing error—a system that grows wiser over time in the exact sense that its sound, reusable structure only ever increases.

AI USE STATEMENT

This paper was written with assistance from Claude (Anthropic). The AI assistant was used for drafting and editing prose for clarity, checking mathematical notation consistency, and formatting. All technical content, definitions, theorems, proofs, and claims are the sole work of the human authors, who verified all content for accuracy.

ETHICS STATEMENT

This work presents a theoretical framework for certified LLM inference. It does not involve human subjects, sensitive data, or dual-use concerns. The framework is designed to increase the reliability and trustworthiness of AI systems by making verification guarantees explicit and computable.

REPRODUCIBILITY STATEMENT

All definitions, theorems, and proofs are stated in full in the main text and Appendix A. The framework is mathematical and does not require code to reproduce. The amortization bounds (Proposition 1), soundness theorem (Theorem 1), and upward-closure result (Proposition **??**) are self-contained and can be verified from the stated assumptions.

REFERENCES

[1] G. Magarshak. Probabilistic Language Tries: A Unified Structure for Compression, Decision, and Computational Reuse. arXiv:2604.06228, 2026.

[2] G. Magarshak. LAWS: Learning from Actual Workloads Symbolically. arXiv:2605.04069, 2026.

[3] G. Magarshak. Generalized LAWS: Trust as a Computable Property of LLM Inference. 2026.

[4] G. Magarshak. Grokers: Bottom-Up Inductive Comprehension and Write-Time Intelligence over Typed Knowledge Graphs. 2026.

[5] A. Gupta and I. S. Mumick. Maintenance of Materialized Views: Problems, Techniques, and Applications. IEEE Data Eng. Bull. 18(2), 1995.

[6] P. Lewis et al. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. NeurIPS, 2020.

[7] Y. Bai et al. How Do AI Agents Spend Your Money? Analyzing and Predicting Token Consumption in Agentic Coding Tasks. arXiv:2604.22750, 2026.

A Full Proofs

[Full proofs of all theorems to be included here.]

B Worked Examples

[Detailed worked examples across code, games, mathematics, and data systems to be included here.]

C Amortization Across Agents

[Multi-agent amortization analysis and content-addressed citation protocol to be included here.]