THE FRONTIER IS NEVER UNDERSTOOD

Anonymous authors
Paper under double-blind review

ABSTRACT

Whether a model can be completely understood by a model no more capable than itself has been asked informally for years and settled neither way, because neither term has been tied to what a system may actually spend. We fix both to a declared deployment budget: a system is *frontier* at that budget if nothing within the budget strictly dominates it, and an interpreter *understands* a system if it retains that system's capabilities and answers a fixed specification of mechanism questions about it. The question then has a two-case answer. Either a frontier system already answers its own mechanism questions, or nothing within its budget understands it, and whatever does strictly dominates it and takes its place; the most capable system in a cohort is therefore unexplainable from inside that cohort rather than merely unexplained. What the escape costs depends entirely on whether the target is fixed when the interpreter is built or supplied with each request. In the first case the optimal interpreter is a lookup table, at a marginal cost of exactly m stored parameters for an m-answer specification, a tight bound on a construction that explains nothing. In the second the parameter cost vanishes and the whole cost falls on the inference deadline, where we bound it and measure it on a transformer across six standard mechanism queries: the factor ranges from one to 2^{LH}, and the number of times understanding can be iterated from about eight to fewer than one. The obstruction reaches only overseers that retain their target's capabilities, leaving weaker-evaluator schemes untouched; where it does reach, conditioning deployment on understanding fixes a permanent gap between the most capable system built and the most capable one deployed.

1 INTRODUCTION

Whether a model can be fully understood by a model no more capable than itself has close formal predecessors and no settled answer. Fallenstein & Soares (2015) argue that predicting a more capable successor in full detail would already confer the capability being predicted, and Wolpert (2008) proves that inference devices cannot strongly infer themselves or one another. Both concern prediction and self-reference; neither asks what a fixed deployment budget permits, and neither bears on which system a laboratory may ship. This paper asks that question, proves the version of the claim that is true, identifies the versions that are false, and bounds the cost.

Two definitions carry the argument. Frontier status is maximality within a declared resource bound, so an instrumented copy of a target is a case the dichotomy classifies rather than an exception to it. Understanding is capability retention plus mechanism coverage, and Corollary 9 narrows this further: retention alone drives the obstruction. The separation follows in two lines from these, and yields a partition of oversight architectures into those that incur the cost and those that do not. Sections 4 to 6 then ask how large that cost is and in which resource.

The obstruction is never informational. Answers to mechanism questions are functions of the target's weights, so given unbounded online computation any system holding those weights produces them. No information is missing anywhere; what can be missing is the budget to compute or store the answers under a real deadline. The analysis is therefore about resources, and any version of the claim phrased in terms of what a model can in principle know is false.

The definition of understanding decides the answer. Two request protocols are available, and they give opposite answers. If an interpreter need only answer about the single target it ships with,

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

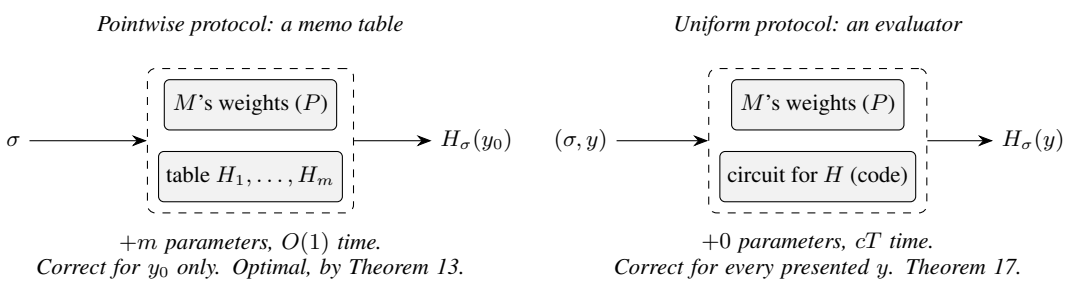

Figure 1: The two request protocols. Left, the target is a fixture of the deployment, answers can be precomputed and stored, the marginal cost is exactly m parameters, and the artefact explains nothing. Right, the target is an argument of the request, precomputation is impossible, no extra parameters are needed, and the whole cost is the online factor c.

the optimum is that target plus a table of precomputed answers and the marginal cost is exactly the number of answers, a bound we show is tight. A table explains nothing, so that protocol does not express understanding. If instead the target arrives with the request and the interpreter must answer for any member of the family, the table fails and the parameter cost falls to zero, leaving the entire cost in the deadline.

A familiar version of the same distinction. Readers with a programming-languages background will recognise the two protocols. A pointwise interpreter is a *memoization table* for one program; a uniform interpreter is an *evaluator*, taking the program as an argument. The quantity c we bound throughout is then the familiar *interpretation overhead* of an evaluator relative to running the program directly, and Figure 1 shows both constructions.

Contributions.

1. **The cohort maximum is unexplained** (Theorems 5 and 8). For a system maximal within a resource bound, either it covers its own mechanism specification or nothing within the bound completely understands it. The most capable existing system is therefore not merely unexplained but unexplainable from inside the cohort it leads, since anything supplying the explanation takes its place at the top. Proposition 10 shows both alternatives are occupied, so the dichotomy is not a universal claim in disguise.
2. **A deployment rule with a construction discipline attached** (Theorem 25, Proposition 24). Conditioning deployment on understanding forbids deploying the cohort maximum, and the gap this forces is structural rather than chosen. Withholding the newest system buys nothing unless it was built as an interpreter of the one below it, which no current pipeline does.
3. **Retention, not completeness, is the operative hypothesis** (Corollary 9), which partitions oversight architectures into those that incur the cost and those that do not.
4. **A degeneracy theorem** (Theorem 13). Under the pointwise protocol a lookup table attains the optimum, at a marginal parameter cost of exactly m, so that protocol does not express understanding. The degeneracy follows from the unconditional upper bound alone; every hypothesis beyond the accounting model is confined to the matching constant.
5. **Zero parameter cost under uniformity, and bounds on what replaces it** (Theorem 17, Theorem 20, Propositions 21 and 22). When the target is presented at request time a complete interpreter needs no stored parameters beyond the target's own, and the escape is paid in online time alone, bounded below by a degree argument, below again by a superpolynomial bound conditional on hardness, and above by a constant factor for derivative-type queries.
6. **Measurements spanning the regimes** (Section 6). We measure c on a closed-form instance and on a transformer across six standard mechanism queries, confirming the predicted constant factor for derivative-type queries at $3.07\times$, and, since a specification costs the maximum over its members, locating the boundary at which the ladder of Corollary 23 collapses to a single level.

7. **A logarithmic ladder** (Corollary 23). Successive levels of understanding are logarithmic in the latency budget, and Proposition 30 shows decomposition across sub-frontier interpreters redistributes the cost without avoiding it.

Scope. The bounds apply to one demanding contract. Partial explanation, supervision by weaker evaluators, self-knowledge, and model compression fall outside it, as does the question of whether a current model meets any hypothesis here. Section 8 and Appendix F place each.

2 SETUP

2.1 RESOURCES AND DEPLOYMENT CLASSES

A *resource vector* is $\mathcal{R} = (P, T, L)$: stored real-valued parameter coordinates, an online inference deadline, and a bound on discrete deployment code. Resource vectors are ordered componentwise, so escaping a bound means exceeding it in at least one coordinate, and Sections 4 and 5 determine which.

Let $\mathcal{C}(\mathcal{R})$ be the class of admissible deployments with resource vector at most $\mathcal{R}$. A deployment comprises an architecture, routing, request protocols, decoders, and any verification machinery, all charged: every target-dependent numerical constant to P, including constants outside nominal neural weights, and all discrete program structure, target-dependent or not, to L. Target-dependent external memory is charged to the combined system, so a target's weights may not be supplied as an uncharged second parameter vector. One stored parameter vector serves all requests, which may arrive separately. Offline selection of that vector is unrestricted, so the results constrain what can be deployed and not the process producing it.

2.2 SPECIFICATIONS, CAPABILITY, AND TOLERANCE

Fix a finite universe $\mathcal{S}$ of operational task specifications. Each $\sigma \in \mathcal{S}$ carries a request protocol and a declared tolerance, and a deployment satisfies σ when it returns an answer within that tolerance, within its own deadline T, on every admissible request. Satisfaction therefore never requires exact equality of real numbers, and it is always relative to the deadline. Write $\mathrm{Cap}(A) \subseteq \mathcal{S}$ for the specifications deployment A satisfies. Capability belongs to a single deployment and does not pool: a capability available only by running two systems that cannot be co-deployed within the budget is in Cap of neither. Define

$$A \succeq B \iff \mathrm{Cap}(B) \subseteq \mathrm{Cap}(A), \qquad A \succ B \iff \mathrm{Cap}(B) \subsetneq \mathrm{Cap}(A).$$

Building tolerance into $\mathcal{S}$ keeps retention sets open rather than varieties; no result below depends on exact equality of reals.

2.3 RESOURCE-BOUNDED FRONTIERNESS

Definition 1 (Resource-bounded frontier)**.** M is *$\mathcal{R}$-frontier* if $M \in \mathcal{C}(\mathcal{R})$ and no $A \in \mathcal{C}(\mathcal{R})$ satisfies $A \succ M$.

This is the central primitive. It asserts the budget is tight against every alternative use of it, a maximality claim rather than one about numerical rank, and is therefore untouched by evidence that trained networks occupy low-dimensional parameter subspaces (Li et al., 2018; Aghajanyan et al., 2021).

Lemma 2 (Existence)**.** *If $\mathcal{S}$ is finite and $\mathcal{C}(\mathcal{R})$ is nonempty, an $\mathcal{R}$-frontier system exists.*

Finiteness of $\mathcal{S}$ is load-bearing: with an infinite specification universe, ascending chains of capability sets need not have upper bounds inside $\mathcal{C}(\mathcal{R})$. All proofs are in Appendix A.

2.4 MECHANISM SPECIFICATIONS

Let $(B_y)_{y \in U}$, with $U \subseteq \mathbb{R}^p$ open and connected, be a family of targets in local coordinates, and let $M = B_{y_0}$ be the target of interest. A *mechanism specification* is a finite set $\mathcal{H} \subseteq \mathcal{S}$ of m

question types, fixed before any candidate interpreter is evaluated and judged against an external mathematical ground truth rather than against what an interpreter reports. Interchange interventions in the sense of causal abstraction (Geiger et al., 2021) are the canonical instance. Nothing requires $\mathcal{H}$ to be real-valued, continuous, or analytic, so discrete and categorical answers are admitted. Write $H : U \to \mathbb{R}^m$ for the ground-truth answer map.

Definition 3 (Complete understanding)**.** Deployment A *completely understands* M if $A \succeq M$ and $\mathcal{H} \subseteq \mathrm{Cap}(A)$.

Neither additional resources nor strict superiority is built in; both are proved. We write $\mathcal{H}$ for the m question types themselves; choosing a request protocol instantiates them as a specific subset of $\mathcal{S}$, and Sections 4 and 5 show that the choice determines everything. The formal treatment takes mechanism answers to be properties of the weights, $H : U \to \mathbb{R}^m$; queries additionally conditioned on a runtime input only strengthen the case against lookup tables.

2.5 Retention and coverage

The two clauses of Definition 3 stand in a definite relation to each other. Call $\mathcal{H}$ *separating* if the answer map H is injective on the family, the minimal formal content of *complete*: a specification giving two systems with different mechanisms the same answers has not pinned the mechanism down.

Proposition 4 (Coverage implies retention)**.** *Let $\mathcal{H}$ be separating and let A satisfy $\mathcal{H}$ under the uniform protocol. Then the answers A returns determine the presented target exactly. If A additionally stores an encoding of M's coordinates within its parameter budget and holds the family's evaluation map as deployment code, then* $\mathrm{Cap}(M) \subseteq \mathrm{Cap}(A)$.

The converse fails, and trivially: a bit-identical copy of M has $\mathrm{Cap}(A) = \mathrm{Cap}(M)$, so in horn *(b)* it retains everything and satisfies no $\sigma \in \mathcal{H} \setminus \mathrm{Cap}(M)$. The contract is therefore asymmetric rather than circular. A separating mechanism account already fixes the target, so retention is what follows once one grants that an account nobody can act on is not an account; coverage is the part retention never supplies.

3 The interpreter dichotomy

With frontierness and understanding both fixed, the question of whether a frontier system can be understood at its own budget has an answer, and the answer has exactly two cases. The whole of this section rests on Definitions 1 and 3 and the finiteness of $\mathcal{S}$.

Theorem 5 (Interpreter dichotomy)**.** *Let M be $\mathcal{R}$-frontier and $\mathcal{H} \subseteq \mathcal{S}$. Exactly one of the following holds.*

(a) *Self-coverage. $\mathcal{H} \subseteq \mathrm{Cap}(M)$: the system answers its own mechanism questions within its own budget.*

(b) *Escape. No $A \in \mathcal{C}(\mathcal{R})$ completely understands M.*

Corollary 6 (Escape)**.** *If M is $\mathcal{R}$-frontier and $\mathcal{H} \not\subseteq \mathrm{Cap}(M)$, every complete interpreter of M lies outside $\mathcal{C}(\mathcal{R})$ and exceeds $\mathcal{R}$ in at least one coordinate.*

Corollary 7 (The interpreter joins the frontier)**.** *Assume the hypotheses of Corollary 6. Then any complete interpreter A of M satisfies $A \succ M$. If in addition M is maximal in a cohort $\mathcal{N}$ fixed before A is introduced, then A is maximal in $\mathcal{N} \cup \{A\}$ and M is not. Completely understanding a frontier system produces a new frontier system.*

This is the sense in which understanding advances the frontier, and it concerns capability sets rather than benchmark scores: A is maximal because it retains everything M does and answers a question M cannot, not because it wins unrelated tasks.

Theorem 8 (The cohort maximum is unexplained)**.** *Let $\mathcal{N}$ be a finite nonempty set of existing deployments and let M be maximal in $\mathcal{N}$ under $\succeq$. If $\mathcal{H} \not\subseteq \mathrm{Cap}(M)$, then no member of $\mathcal{N}$ completely understands M.*

The step from understanding to dominance is conditional, not definitional. It runs only when some $\sigma \in \mathcal{H}$ lies outside $\mathrm{Cap}(M)$: a fact about a system and its specification that no definition supplies, that Proposition 10 shows can fail, and that Theorem 20 decides from the deadline. In horn *(a)* the same definitions hold and the conclusion does not follow, since M understands itself without dominating itself.

No resource bound appears in Theorem 8. At any moment the most capable existing system either answers its own mechanism questions or is completely understood by nothing that exists, and the second case is not a shortfall that further effort removes: by Corollary 6 the explanation is unavailable at that budget, and by Corollary 7 anything supplying it takes the target's place at the top. Which case holds is a property of the system rather than a stipulation. Theorem 20 gives a sufficient condition for the second by bounding what a target can compute inside its own deadline, and Section 6.2 measures the relevant quantity on a transformer.

Corollary 9 (Retention is the operative hypothesis)**.** *Let M be $\mathcal{R}$-frontier and let $\mathcal{H}' \subseteq \mathcal{S}$ be* any *nonempty set of mechanism questions, complete or not. If some $\sigma \in \mathcal{H}'$ lies outside $\mathrm{Cap}(M)$, then no $A \in \mathcal{C}(\mathcal{R})$ both retains $\mathrm{Cap}(M)$ and satisfies $\mathcal{H}'$.*

The proof of Theorem 5 appeals to no property of $\mathcal{H}$, so what drives the obstruction is *retention*. An overseer reproducing everything its target does and answering one mechanism question it cannot already escapes the budget, and schemes whose evaluators do not retain target capabilities (Irving et al., 2018; Christiano et al., 2018; Burns et al., 2023) fall outside the hypothesis for that reason. Completeness reappears below only because the *cost* results require a finite specification to count.

Proposition 10 (Both horns occur)**.** *Let M apply a matrix $W \in \mathbb{R}^{n\times n}$, with task mode the action of W on presented vectors at cost $2n^2 - n$ operations. If $\mathcal{H} = \{\mathrm{tr}\, W\}$, a trace mode costs $n - 1$ additions and $O(1)$ code, so the deployment satisfies $\mathcal{H}$ inside its own deadline and M lies in horn* (a)*. Horn* (b) *is occupied by any target whose mechanism mode is a branch-free arithmetic program admitting at most t multiplications faced with a specification of degree $D > 2^t$, which Theorem 20 excludes.*

Both alternatives are occupied, so which horn holds is fixed by the specification and the deadline together rather than by the target alone: a laboratory that writes down what it wants explained is choosing which side its system falls on. The determinant of Section 6.1 is not a horn *(b)* witness by the degree route, since a task mode performing n^2 multiplications already admits degrees far above n; there the obstruction is the $\Theta(n^3)$ evaluation cost against an $O(n^2)$ deadline.

Remark 11 (Instrumented targets)**.** Consider the instrumented identity interpreter: keep M's weights and add a mode running one extra forward pass with an activation patch applied, leaving parameters unchanged and task behaviour bit-identical. This is horn (a). If it fits $\mathcal{R}$ it either coincides with M up to routing, so $\mathcal{H} \subseteq \mathrm{Cap}(M)$, or strictly dominates M within $\mathcal{R}$, contradicting $\mathcal{R}$-frontierness. Cheap mechanism questions therefore place the burden on a claim about M that Theorem 20 and Section 6 make checkable.

4 The pointwise definition is degenerate

Theorem 5 says an interpreter must leave the budget. It does not say by how much, and the answer turns on a detail Definition 3 deliberately left open: what a mechanism request is allowed to name. Two choices are available. This section takes the narrower one, under which an interpreter answers only about the target it ships beside, and shows that it collapses.

Definition 12 (Pointwise protocol)**.** Under the *pointwise* protocol, a mechanism request names a question type $\sigma \in \mathcal{H}$ and nothing else. The interpreter is deployed alongside a fixed target and must return $H_\sigma(y_0)$ for that target.

Theorem 13 (Pointwise marginal cost is exactly m)**.** *Fix the pointwise protocol and a finite $\mathcal{H}$ with m scalar answers.* (i) *If a design retains $\mathrm{Cap}(M)$ with q parameters, then appending m stored coordinates gives a complete interpreter with $q+m$ parameters, $T+\tau$ deadline and $L+O(1)$ code, where τ is the cost of emitting a stored coordinate. Taking the target itself as that design, $P + m$ suffices.* (ii) *Every retaining design has $q \geq p$, and no design with at most $p + m - 1$ parameters succeeds for almost every target, under a rank condition on the mechanism residual stated and proved as Theorem 28 in Appendix B. The* marginal *cost of complete mechanism coverage, over and above whatever retention itself costs, is therefore exactly m.*

The construction is a lookup table appended to an unmodified copy of the target, and offline selection is unrestricted, so the builder may compute $H(y_0)$ by any means and store the result. Half *(i)* turns on finiteness of $\mathcal{H}$ alone, and it establishes the degeneracy by itself: the table exists, so the protocol admits an optimum that explains nothing. Half *(ii)* fixes the constant, and carries the rank hypothesis of Appendix B. Both halves are stated in the same currency, so the quantity they pin down is the increment m rather than an absolute parameter count; Appendix D expands on this.

Remark 14 (The pointwise optimum)**.** The interpreter attaining Theorem 13 stores answers it never computes, says nothing about any target but the one it shipped with, and returns identical outputs if that target is replaced by an unrelated system with the same task responses, so the pointwise protocol does not express understanding. Requiring one design to work across a family leaves it intact, since per-target offline selection is such a design and still costs m. What excludes it is making the target an argument of the question, which is the protocol of Section 5.

5 UNIFORM UNDERSTANDING

The table of Theorem 13 survives only because the target is fixed when the interpreter is built. Making the target an argument of the request removes that possibility, and with it the entire parameter cost.

Definition 15 (Uniform protocol)**.** Under the *uniform* protocol, a mechanism request names a question type $\sigma \in \mathcal{H}$ together with a presented target $y \in U$, supplied as request input. Specification σ is satisfied only if the deployment returns $H_\sigma(y)$ within tolerance and within the deadline for *every* admissible request. Request input is not a stored constant and is charged to neither P nor L; reading it is charged to the deadline.

This is what interpretability practice already means by a mechanism account: a claim about how a class of systems computes something, not a dossier on one checkpoint. It leaves $\mathcal{S}$ finite, since $\mathcal{H}$ remains m question types with the quantification over targets inside each satisfaction condition.

Proposition 16 (The table fails uniformity)**.** *Let A be any deployment whose mechanism mode ignores the presented target. If for some $\sigma \in \mathcal{H}$ the oscillation of H_σ over U exceeds twice the tolerance declared for σ, then $\sigma \notin \mathrm{Cap}(A)$ under the uniform protocol. In particular the interpreter of Theorem 13 does not completely understand M uniformly.*

Every statement of Section 3 survives verbatim, since Theorem 5 depends only on Cap and not on which protocol generated it, and horn (a) becomes harder to satisfy, since self-coverage now requires answering about presented targets rather than about oneself.

5.1 THE PARAMETER COST VANISHES

Theorem 17 (Uniform upper bound)**.** *Suppose each H_σ is computable from presented weights by an arithmetic circuit of size S_σ and online cost at most $c_\sigma T$, and set $c = \max_\sigma c_\sigma$. Then M admits a complete uniform interpreter in*

$$\mathcal{C}\big(P,\ \max(T, cT),\ L + \textstyle\sum_\sigma S_\sigma\big).$$

No stored parameters beyond M's own are required. The deadline takes the maximum because requests arrive separately and each must be met individually; the code bound takes the sum because every circuit must be stored.

Corollary 18 (The operative factor is the most demanding question)**.** *For a mechanism specification $\mathcal{H}$, the factor governing Corollary 23 is $c(\mathcal{H}) = \max_{\sigma\in\mathcal{H}} c_\sigma$. Adding a cheap question to a specification never lowers its cost, and a specification is as expensive as its single most demanding member.*

A partial interpreter chooses its queries and can sit at the cheap end; a complete one inherits the maximum over a specification it did not choose.

Corollary 19 (The currency of the escape)**.** *Under the uniform protocol and the hypotheses of Theorem 17, the escape guaranteed by Corollary 6 need never be paid in P: it can always be accommodated in the deadline T and the discrete code bound L without expanding parameter storage.*

Under the uniform protocol, then, understanding never requires more weights: circuit structure is discrete and charged to L, the presented target is request input rather than storage, and the stored vector need not grow. The deadline binds, and the rest of the paper bounds its growth.

Distributing $\mathcal{H}$ across several interpreters, none frontier-capable, so that no single system dominates M, redistributes the cost rather than avoiding it. An ensemble is itself an admissible deployment, with parameters, code and deadline summed or maximised over its components, so Theorem 5 applies to it directly: either the ensemble fails to retain $\mathrm{Cap}(M)$, which places it outside Definition 3 as partial interpretation, or it retains and leaves the budget in aggregate. Proposition 30 in Appendix F states this and proves it. Decomposition secures that no *individual* component is frontier-capable, which bears on containment and not on the aggregate.

5.2 Lower bounds on the time cost

Theorem 20 (Degree bound)**.** *If H_σ is a polynomial of total degree D in the presented weights, then any branch-free arithmetic interpreter computing it uses at least $\lceil \log_2 D \rceil$ multiplications. Consequently, if M's own online budget admits at most t multiplications with $2^t < D$, then $\sigma \notin \mathrm{Cap}(M)$ and horn (a) fails for σ.*

Proposition 21 (Hardness-conditional bound)**.** *If satisfying σ is NP-hard in the presented weights, no interpreter with a polynomial-time online procedure satisfies σ unless $\mathrm{P} = \mathrm{NP}$, so the deadline gap is superpolynomial under that assumption. Existing hardness results for post-hoc explanation queries and for circuit discovery supply instances (Barceló et al., 2020; Adolfi et al., 2025).*

Proposition 22 (Derivative-type queries are cheap)**.** *If H_σ consists of the partial derivatives of a* scalar-valued *function the target already computes, then by the Baur–Strassen theorem all of them are obtainable at a constant factor of the target's own arithmetic cost (Baur & Strassen, 1983; Baydin et al., 2018), so $c = O(1)$ for such σ. For a k-output map the same argument costs k reverse sweeps, which is why the layer Jacobian of Table 1 sits at $c \approx d$ rather than $O(1)$.*

Propositions 21 and 22 delimit the regime where the obstruction is substantive: for gradient-like and ablation-like queries the escape is a constant factor and the ladder of Section 7 is long, while for queries in the hard classes it is superpolynomial and the ladder is short. Theorem 20 applies unconditionally but sits far from the upper bound, and closing that gap for natural interpretability queries is the main open problem here.

6 Measuring the online cost

Corollary 19 puts the whole cost in the deadline, so how large the obstruction is has become a question about a single measurable number. No experiment on deployed frontier models is reported here. We measure c, the factor in Theorem 17 and Corollary 23, in two settings: a closed-form instance where every quantity is available analytically, and a transformer where the mechanism questions are the ones interpretability research actually asks. Online cost depends on shapes and the query, not on weight values, so randomly initialised networks are used throughout.

6.1 A closed-form instance

A linear target applying $W \in \mathbb{R}^{n \times n}$, with task mode the action of W on presented vectors and mechanism question $\det W$, is the one case where every quantity in the theory is analytic: $p = n^2$, the degree $D = n$, both costs in closed form, and the certificate of Theorem 20 available exactly. Appendix E gives the derivation and measurements; the ratio runs from 17 at $n = 64$ to 78 at $n = 2048$, far above the $\lceil \log_2 n \rceil$ bound, as expected of a bound counting multiplications alone. The determinant is not a question interpretability research asks, which is what the next subsection supplies.

6.2 A transformer and the queries interpretability actually asks

We instrument a transformer with $L = 4$ layers, $H = 4$ heads, $d_{\mathrm{model}} = 128$, $d_{\mathrm{ff}} = 512$, and sequence length $S = 64$, implemented in PyTorch and run single-threaded on CPU. Autograd is checked with `torch.autograd.gradcheck` in double precision before any timing is taken.

Table 1: Online cost ratio c for standard mechanism queries on a transformer target, median over repeats, with the ladder depth $\log_c B$ each implies at $B = 10^4$. The rightmost column is *projected*, not measured: the pass count for the same query at a deployment-scale shape ($L = 48$, $H = 96$, $d = 12288$, $S = 8192$). Every row is charged at standalone online cost, so the interchange row runs both the source and the patched pass. The exhaustive row is the only projected entry, obtained from the measured forward cost.

mechanism query	passes	measured c	depth at $B = 10^4$	at scale
Interchange intervention	2	1.97	13.6	2
Input-attribution gradient	$O(1)$	3.07	8.2	$O(1)$
Head-ablation sweep	LH	16.4	3.3	4,608
Layer Jacobian (forward mode)	d	128	1.9	12,288
Causal-tracing grid	LS	248	1.7	393,216
Minimal head subset (exhaustive)	2^{LH}	6.6×10^4	0.83	2^{4608}

Task mode is one forward pass, measured at 1.90 ms, and each query is charged at standalone online cost. Each mechanism query is one of the standard operationalisations of a mechanism claim, and each is charged at its full online cost. Table 1 reports the results; Figure 2 in Appendix E plots them against the three regimes the theory predicts.

The measurements confirm the trichotomy and locate its boundaries. The measured gradient ratio is 3.07, consistent with the constant-factor bound Baur–Strassen guarantees, and interchange intervention costs two passes. Both are single questions rather than specifications, and by Corollary 18 a specification costs the maximum over its members, so these rows give a floor: an $\mathcal{H}$ of interventions and gradients alone is cheap, while an $\mathcal{H}$ rich enough to be complete inherits its most demanding member. Cost is set by the richness of the specification rather than the size of the target, sweeps and tracing grids costing one to two orders of magnitude more and rising to 10^3–10^5 passes at deployment shapes. The consequence for Corollary 23 is direct: at a tolerated latency ratio of 10^4, generous for anything running alongside a production system, a gradient-level specification supports about eight successive levels at this scale, a sweep-level specification two, and an exhaustive-search specification none. At the deployment shape the projected pass counts are larger by three orders of magnitude, and a sweep-level specification supports no level at all.

Small c and the separation. The factor c enters no hypothesis of Theorem 5: frontierness makes the budget tight, so 3.07 escapes it exactly as 248 does, and a deadline with slack enough to absorb its own mechanism mode would put M in horn (a) rather than outside the dichotomy. These measurements bound the price of the escape, not its occurrence, and neither setting is a frontier generalist.

7 THE UNDERSTANDING LADDER

A measured c answers a question the dichotomy raises but does not settle: how often the construction can be repeated. Iterating Corollary 7 along a chain of successive interpreters gives strictly growing capability sets and strictly growing budgets (Corollary 26, Appendix A).

Corollary 23 (Logarithmic ladder)**.** *Under the uniform protocol, suppose each rung is paid in the deadline at a factor of at least $c > 1$, as Corollary 19 requires when parameters do not grow. Then a tolerable latency ratio $B = T_{\max}/T_0$ supports at most $\log_c B$ successive levels of complete understanding.*

The corollary assumes a fixed factor per rung; if richer targets demand richer specifications then c grows by Corollary 18 and the true depth is smaller, so the bound errs in the direction that matters. It cuts both ways: Theorem 17 guarantees another rung at a bounded factor, while a multiplicative cost exhausts any latency budget after logarithmically many levels.

Proposition 24 (Surplus is not coverage)**.** *$\mathcal{R}_{i+1}$ exceeding $\mathcal{R}_i$ does not imply $\mathcal{H}_{M_i} \subseteq \mathrm{Cap}(M_{i+1})$. A more capable successor is not an interpreter of its predecessor unless it was built to the specification.*

Proposition 24 is what separates the rule below from the familiar instinct to stay a generation behind. Successive model generations are built to be more capable, not to cover their predecessors' mechanisms, so a lab that withholds its newest system while having trained it for capability obtains no coverage of the system it ships. The ladder describes a construction discipline, and no observed release sequence satisfies it.

8 What follows for deployment

Theorem 8 constrains any policy that conditions deployment on understanding.

Theorem 25 (Second-to-last deployment)**.** *Consider the policy: deploy A only if some existing system completely understands A. Under this policy and Theorem 8, a maximal element of the existing cohort is not deployable unless it is self-covering; consequently, when no maximal element is self-covering, the deployable set is contained in the non-maximal part of the cohort.*

The rule is not the familiar instinct to stay a generation behind, and two conditions separate it from that instinct. First, the gap it forces is structural rather than chosen: by Theorem 8 the maximal element of a cohort is unexplainable by anything in the cohort, so no additional effort at fixed budget closes the distance between the best existing system and the best deployable one. Second, by Proposition 24 withholding the top generation buys nothing unless that generation was built as an interpreter of the one below it: the rule is "build each generation to explain its predecessor, and retain the newest as an instrument," not "ship one generation behind." A lab that trains its newest system for capability and withholds it obtains no coverage of what it ships. No current pipeline works this way, so Theorem 25 is a counterfactual proposal rather than a description of present practice. Third, by Corollary 9 the rule requires no premise about completeness. It requires only that the overseer retain the target's capabilities and answer some mechanism question the target cannot, so an oversight scheme with a capability-retaining overseer lies inside the ladder whatever else it demands, and one built on a weaker overseer (Irving et al., 2018; Christiano et al., 2018; Burns et al., 2023) lies outside it. Appendix F treats containment, the live-monitor hazard, and cost, the last of which Corollary 19 makes a latency cost rather than a capacity one.

Scope. Partial explanation is untouched, as is offline analysis followed by distillation, on which Theorem 25 depends. Components never co-deployed never place the union of capabilities in one deployment, so no accounting over deployments reaches them.

9 Related work

The two predecessors named in Section 1 are the closest; Appendix G treats them and the rest in full. Barceló et al. (2020) and Adolfi et al. (2025) show specified explanation queries can be intractable, bounding the cost of *computing* an explanation where we bound the resources of *deploying* one with offline selection unrestricted; they meet us at Proposition 21, which supplies the instances where our time gap is superpolynomial. Maurer et al. (2016) and Elhage et al. (2022) show abilities need not occupy disjoint weights, and Baur & Strassen (1983) obtains derivatives at constant-factor overhead; the accounting admits both, and Proposition 22 turns the latter into an upper bound rather than an obstacle. Li et al. (2018) and Hinton et al. (2015) argue against inferring exhaustion from parameter count, which is why Definition 1 avoids saturation. Geiger et al. (2021) supply the route to specifying mechanism claims behind $\mathcal{H}$, and Definition 15 formalises what that framework assumes. Jacovi & Goldberg (2020), Bills et al. (2023), and Lindsey (2025) document faithfulness failures and partial self-knowledge, bearing on horn *(a)*.

10 Conclusion

Fix a budget and a system maximal within it. Either it answers its own mechanism questions, or nothing within the budget both retains its capabilities and answers them, and whatever does

joins the frontier. Retention rather than completeness drives this, and an ensemble is charged as an ensemble, so neither a weaker evaluator nor a decomposition into sub-frontier components alters the conclusion. Answering only about the target one ships with is optimally a lookup table; presenting the target with the question drives the parameter cost to zero and leaves everything in the deadline. The most capable system a laboratory possesses is thus the one it understands least, and conditioning deployment on understanding does not narrow that gap but fixes it in place.

Reproducibility statement

All theoretical claims are proved in Appendix A, with Appendix D listing the assumptions each result uses. The measurements of Section 6 are produced by two self-contained PyTorch scripts included in the supplementary material, using a fixed random seed, a single CPU thread, no accelerator, and no learned components. The transformer script asserts `torch.autograd.gradcheck` in double precision before reporting any timing. Both run in a few minutes and report median wall-clock, and both emit the raw JSON from which the tables are transcribed.

Ethics statement

This paper is theoretical and uses no human subjects, no personal data, and no trained models. Section 8 draws a governance conclusion, and its scope is bounded by Corollary 9: the obstruction attaches to capability-retaining overseers, so the weaker-evaluator research programs are unaffected, as is offline analysis followed by distillation, on which the deployment rule of Theorem 25 depends. The results should not be read as an argument that oversight is infeasible, and Theorem 25 describes a construction discipline no current pipeline follows rather than a verdict on present practice.

Use of large language models

A large language model was used as a general-purpose assistant: for prose editing, for adversarial critique of draft arguments, for drafting proof write-ups subsequently checked by the authors, and for producing the measurement scripts of Section 6. The research direction, the central claims, and the verification of every stated result are the authors', who take full responsibility for the content.

References

Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. Sanity checks for saliency maps. In *Advances in Neural Information Processing Systems*, 2018.

Federico Adolfi, Martina G. Vilas, and Todd Wareham. The computational complexity of circuit discovery for inner interpretability. In *International Conference on Learning Representations*, 2025.

Armen Aghajanyan, Sonal Gupta, and Luke Zettlemoyer. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In *Proceedings of ACL-IJCNLP*, pp. 7319–7328, 2021.

Pablo Barceló, Mikaël Monet, Jorge Pérez, and Bernardo Subercaseaux. Model interpretability through the lens of computational complexity. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Walter Baur and Volker Strassen. The complexity of partial derivatives. *Theoretical Computer Science*, 22:317–330, 1983.

Atilim Gunes Baydin, Barak A. Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning: A survey. *Journal of Machine Learning Research*, 18(153):1–43, 2018.

Steven Bills, Nick Cammarata, Dan Mossing, Henk Tillman, Leo Gao, Gabriel Goh, Ilya Sutskever, Jan Leike, Jeff Wu, and William Saunders. Language models can explain neurons in language models, 2023. OpenAI research publication.

Collin Burns, Pavel Izmailov, Jan Hendrik Kirchner, Bowen Baker, Leo Gao, Leopold Aschenbrenner, Yining Chen, Adrien Ecoffet, Manas Joglekar, Jan Leike, Ilya Sutskever, and Jeff Wu. Weak-to-strong generalization: Eliciting strong capabilities with weak supervision. *arXiv:2312.09390*, 2023.

Paul Christiano, Buck Shlegeris, and Dario Amodei. Supervising strong learners by amplifying weak experts. *arXiv:1810.08575*, 2018.

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, et al. Toy models of superposition. *arXiv:2209.10652*, 2022.

Benja Fallenstein and Nate Soares. Vingean reflection: Reliable reasoning for self-improving agents. Technical Report 2015-2, Machine Intelligence Research Institute, 2015.

Benja Fallenstein, Jessica Taylor, and Paul F. Christiano. Reflective oracles: A foundation for classical game theory. *arXiv:1508.04145*, 2015.

Atticus Geiger, Hanson Lu, Thomas Icard, and Christopher Potts. Causal abstractions of neural networks. In *Advances in Neural Information Processing Systems*, 2021.

Shervin Ghasemlou. Verifiable explanations cannot be much smaller than the behavior they explain. In *ICML Mechanistic Interpretability Workshop*, 2026.

Aravind Gollakota, Parikshit Gopalan, Aayush Karan, Charlotte Peale, and Udi Wieder. When does a predictor know its own loss? In *6th Symposium on Foundations of Responsible Computing (FORC)*, volume 329, pp. 22:1–22:22, 2025.

Juris Hartmanis and Richard E. Stearns. On the computational complexity of algorithms. *Transactions of the American Mathematical Society*, 117:285–306, 1965.

John Hewitt and Percy Liang. Designing and interpreting probes with control tasks. In *Proceedings of EMNLP-IJCNLP*, pp. 2733–2743, 2019.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv:1503.02531*, 2015.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, et al. Training compute-optimal large language models. *arXiv:2203.15556*, 2022.

Geoffrey Irving, Paul Christiano, and Dario Amodei. Ai safety via debate. *arXiv:1805.00899*, 2018.

Alon Jacovi and Yoav Goldberg. Towards faithfully interpretable nlp systems: How should we define and evaluate faithfulness? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4198–4205, 2020.

Saurav Kadavath, Tom Conerly, Amanda Askell, et al. Language models (mostly) know what they know. *arXiv:2207.05221*, 2022.

Chunyuan Li, Heerad Farkhoor, Rosanne Liu, and Jason Yosinski. Measuring the intrinsic dimension of objective landscapes. In *International Conference on Learning Representations*, 2018.

Jack Lindsey. Emergent introspective awareness in large language models, 2025. Transformer Circuits research publication.

Andreas Maurer, Massimiliano Pontil, and Bernardino Romera-Paredes. The benefit of multitask representation learning. *Journal of Machine Learning Research*, 17(81):1–32, 2016.

Stuart J. Russell and Devika Subramanian. Provably bounded-optimal agents. *Journal of Artificial Intelligence Research*, 2:575–609, 1995.

Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. In *Advances in Neural Information Processing Systems*, 2018.

Héctor J. Sussmann. Uniqueness of the weights for minimal feedforward nets with a given input-output map. *Neural Networks*, 5(4):589–593, 1992.

David H. Wolpert. Physical limits of inference. *Physica D: Nonlinear Phenomena*, 237:1257–1281, 2008.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. In *Advances in Neural Information Processing Systems*, 2020.

Jiang Zhang, Bing Yuan, and Qian Zhang. Self-reference in large language models: The introspection threshold for recursive self-improvement. *arXiv:2607.04277*, 2026.

A PROOFS

Corollary 26 (Strict chains)**.** *Let $M_0, M_1, \ldots$ satisfy: M_i is $\mathcal{R}_i$-frontier, M_{i+1} completely understands M_i, and $\mathcal{H}_{M_i} \not\subseteq \mathrm{Cap}(M_i)$ at every stage. Then $\mathrm{Cap}(M_0) \subsetneq \mathrm{Cap}(M_1) \subsetneq \cdots$ and each $\mathcal{R}_{i+1}$ exceeds $\mathcal{R}_i$ in at least one coordinate.*

Proof of Corollary 26. Apply Corollary 7 at each stage for the strict inclusion, and Corollary 6 for the resource increase. □

Proof of Lemma 2. The family $\{\mathrm{Cap}(A) : A \in \mathcal{C}(\mathcal{R})\}$ is a nonempty collection of subsets of the finite set $\mathcal{S}$, hence finite, hence has a maximal element under inclusion. Any A attaining it is $\mathcal{R}$-frontier. □

Proof of Theorem 5. Suppose (a) fails, so there is $\sigma \in \mathcal{H} \setminus \mathrm{Cap}(M)$. Let $A \in \mathcal{C}(\mathcal{R})$ completely understand M. Then $\mathrm{Cap}(M) \subseteq \mathrm{Cap}(A)$ by retention, and $\sigma \in \mathrm{Cap}(A)$ while $\sigma \notin \mathrm{Cap}(M)$, so $\mathrm{Cap}(M) \subsetneq \mathrm{Cap}(A)$ and $A \succ M$, contradicting $\mathcal{R}$-frontierness. Hence (b).

Suppose (a) holds. Since $M \succeq M$ and $\mathcal{H} \subseteq \mathrm{Cap}(M)$, the deployment M itself completely understands M, and $M \in \mathcal{C}(\mathcal{R})$, so (b) fails. The alternatives are mutually exclusive and exhaustive. □

Proof of Corollary 6 and Theorem 8. Corollary 6 is horn (b) together with the observation that $A \notin \mathcal{C}(\mathcal{R})$ means $\mathcal{R}_A \not\leq \mathcal{R}$, which under the componentwise order means at least one coordinate strictly exceeds. For Theorem 8, replace $\mathcal{C}(\mathcal{R})$ by $\mathcal{N}$ in the first paragraph above; only maximality of M in the relevant collection is used. □

Proof of Theorem 13. Upper bound. Construct A with stored parameter vector the concatenation of M's P coordinates, unchanged, with m additional coordinates set offline to the values $H(y_0)$. Its architecture is M's plus a routing predicate on request type and a decoder emitting the appropriate stored coordinate, both discrete and $O(1)$ in L. On task requests A executes M's computation on M's coordinates and is bit-identical to M, so it satisfies every specification M satisfies at the same tolerance within T, giving $\mathrm{Cap}(M) \subseteq \mathrm{Cap}(A)$. On mechanism requests A emits the stored answer, correct by construction against the external ground truth, at cost τ; hence $\mathcal{H} \subseteq \mathrm{Cap}(A)$ under the pointwise protocol. Task and mechanism requests arrive separately and a deployment need only meet the deadline per request, so $T + \tau$ suffices. Offline selection is unrestricted, so computing $H(y_0)$ is not charged to the deployment.

Lower bound. Apply Theorem 28 with $s = m - 1$ and $k = m$: no design with at most $p + m - 1$ parameters succeeds for μ-almost every target. □

Proof of Theorem 28. Designs with $q_a < p$ have task images of p-dimensional Lebesgue measure zero by Lemma 27. For any remaining design and $\alpha_0 \in Z_a$, choose k components of R_a with independent differentials at α_0. The implicit function theorem makes their common zero set near α_0 a C^1 submanifold of dimension $q_a - k$, and Z_a is locally contained in it. Second countability gives a countable collection of such neighborhoods. Since $q_a - k \leq p + s - k < p$, Lemma 27 in submanifold charts shows $F_a(Z_a)$ is null. This image contains every target coordinate for which design a solves the joint conditions. The countable union over designs is null, and $\mu \ll \lambda^p$ converts it to a probability-zero event. □

Lemma 27 (Lower-dimensional images). *If $V \subseteq \mathbb{R}^q$ is open, $q < p$, and $f : V \to \mathbb{R}^p$ is C^1, then $f(V)$ is p-dimensional Lebesgue null. The same holds for the image of a C^1 manifold of dimension $q < p$.*

Proof. Cover V by countably many closed cubes in its interior. On each the derivative is bounded, so f is Lipschitz there. Subdividing into $O(h^{-q})$ cubes of side at most h, the images have diameter $O(h)$ and admit a cover by p-dimensional cubes of total volume $O(h^{p-q})$. Letting $h \downarrow 0$ gives nullity on that cube, and a countable union gives the claim. A second-countable manifold has a countable atlas, so the Euclidean argument applies chartwise. □

Proof of Proposition 30. Concatenating the components' stored parameter vectors gives a single vector of length $\sum_i P_i$, and their circuits together with a routing predicate on request type are discrete structure charged to $\sum_i L_i + O(1)$. Since every request is served by running components and returning their combined output, the deadline is the maximum of the component costs under parallel execution and their sum under sequential execution. Hence E satisfies the conditions of Section 2 and is an admissible deployment. Apply Theorem 5 to E: if $\mathrm{Cap}(M) \not\subseteq \mathrm{Cap}(E)$ then E fails the retention clause of Definition 3; otherwise E retains $\mathrm{Cap}(M)$ and satisfies $\mathcal{H}$, so some $\sigma \in \mathcal{H} \setminus \mathrm{Cap}(M)$ lies in $\mathrm{Cap}(E)$, giving $E \succ M$, and $\mathcal{R}$-frontierness of M forces $E \notin \mathcal{C}(\mathcal{R})$. The two branches are exhaustive and mutually exclusive. □

Proof of Corollary 9. Identical to the first paragraph of the proof of Theorem 5, which uses only $\sigma \in \mathcal{H}' \setminus \mathrm{Cap}(M)$, retention, and maximality of M in $\mathcal{C}(\mathcal{R})$. □

Proof of Proposition 4. Injectivity of H makes $H(y)$ determine y on the family, so a deployment returning $H_\sigma(y)$ for every $\sigma \in \mathcal{H}$ and every presented y fixes the target exactly, which is the first claim. A task request supplies a domain input and not the target, so the second claim needs the stored encoding: with it and the evaluation map in code, A reproduces M's response at the declared tolerance on any task request, giving $\mathrm{Cap}(M) \subseteq \mathrm{Cap}(A)$. □

Proof of Proposition 16. Under Definition 15, σ is satisfied only if the deployment returns $H_\sigma(y)$ within tolerance for every admissible request, including requests presenting distinct targets $y \neq y'$. A mechanism mode ignoring the presented target returns the same value on both. By hypothesis there are y, y' with $|H_\sigma(y) - H_\sigma(y')|$ exceeding twice the declared tolerance, so no single value lies within tolerance of both and at least one response fails; hence $\sigma \notin \mathrm{Cap}(A)$. Nonconstancy alone is insufficient: a target-agnostic deployment returning the midpoint of a range narrower than twice the tolerance would satisfy σ. □

Proof of Theorem 17. Construct A with stored parameter vector exactly M's, and architecture M's plus a routing predicate and the circuits computing each H_σ. Task requests are handled by M's computation and are bit-identical to M, giving $\mathrm{Cap}(M) \subseteq \mathrm{Cap}(A)$ within T. A mechanism request supplies y as input; A evaluates the corresponding circuit on y and returns the result, correct for every presented target by construction, within cT by hypothesis. The circuits are discrete structure charged to L at $\sum_\sigma S_\sigma$, and the presented target is request input charged to neither P nor L. Requests arrive separately, so the deadline requirement is $\max(T, cT)$. Hence A completely understands M uniformly and lies in the stated class. □

Proof of Corollary 19. Theorem 17 exhibits a complete uniform interpreter with the same P. By Corollary 6 any complete interpreter exceeds $\mathcal{R}$ in some coordinate, and since P need not increase, the excess occurs in T or L. □

Proof of Theorem 20. Constants and inputs have degree at most one. Addition does not increase maximum degree, and each multiplication at most doubles it, so a branch-free arithmetic program with t multiplications computes a polynomial of degree at most 2^t. Computing a polynomial of degree D therefore requires $2^t \geq D$, that is $t \geq \lceil \log_2 D \rceil$. If M's online budget admits at most t multiplications with $2^t < D$, no such program available to M computes H_σ, and since a deployment must satisfy every specification in Cap within its deadline, $\sigma \notin \mathrm{Cap}(M)$. □

Proof of Proposition 21. A polynomial-time online procedure satisfying an NP-hard specification on every admissible request would decide an NP-hard problem in polynomial time. □

Proof of Proposition 24. Nothing in Definition 3 follows from a comparison of resource vectors. For a witness, let M_{i+1} extend M_i with capabilities on task specifications disjoint from $\mathcal{H}_{M_i}$; then $\mathcal{R}_{i+1}$ exceeds $\mathcal{R}_i$ while $\mathcal{H}_{M_i} \not\subseteq \mathrm{Cap}(M_{i+1})$. □

Proof of Theorem 25. Let M be maximal in the cohort and not self-covering. By Theorem 8 no member of the cohort completely understands M, so the policy forbids deploying M. Applying this to every maximal element gives the second clause. □

Proof of Corollary 23. By Corollary 19 each rung is paid in T or L; under the stated hypothesis $T_{i+1} \geq c\,T_i$, so $T_i \geq c^i T_0$, and $T_i \leq T_{\max}$ forces $i \leq \log_c(T_{\max}/T_0)$. □

B THE MATCHING LOWER BOUND

Half *(ii)* of Theorem 13 rests on the following, which is the nonalignment hypothesis of earlier formulations of this argument expressed as a rank condition on the mechanism residual. It is the strongest assumption used anywhere in this paper, and it supports only the tightness constant for the protocol Section 4 rejects.

Theorem 28 (Dimension lower bound)**.** *Let $F_a : \Omega_a \to \mathbb{R}^p$ and $G_a : \Omega_a \to \mathbb{R}^m$ be a design's task and mechanism response maps, C^1, with $R_a = G_a - H \circ F_a$ and $Z_a = \{\alpha \in F_a^{-1}(U) : R_a(\alpha) = 0\}$. Let the design set be countable and $\mu \ll \lambda^p$. Fix $s \geq 0$ and $k > s$, and suppose every design with $p \leq q_a \leq p + s$ has $\operatorname{rank} DR_a(\alpha) \geq k$ for all $\alpha \in Z_a$. Then μ-almost every target admits no complete interpreter using at most $p + s$ parameters.*

Applying it with $s = m - 1$ and $k = m$ gives half *(ii)*.

C FINITE TOLERANCE AND WHY ITS QUALIFICATIONS ARE NECESSARY

Exact equality of reals is stronger than practical accuracy, and a measure-zero statement does not by itself supply a numerical error threshold. This appendix supplies the missing step for Theorem 28. Let $K \subset U$ be compact with μ supported on K, restrict to finitely many designs each with compact parameter set A_a, and let S_ε be the set of targets admitting an ε-approximate joint witness.

Proposition 29 (Finite-tolerance separation)**.** *If $\mu(S_0) = 0$ then $\lim_{\varepsilon \downarrow 0} \mu(S_\varepsilon) = 0$, so for every $\eta > 0$ there is a strictly positive ε_η with $\mu(S_{\varepsilon_\eta}) < \eta$.*

Proof. Each feasible set in $K \times A_a$ is closed and compact, so its projection onto K is compact; a finite union shows S_ε is measurable. The sets $S_{1/n}$ decrease. If y lies in every $S_{1/n}$, choose witnesses (a_n, α_n); a subsequence uses one fixed design, and compactness supplies a further subsequence converging to $\alpha^* \in A_a$. Continuity gives both exact equalities, so $y \in S_0$. Hence $\bigcap_n S_{1/n} = S_0$, and continuity from above yields the limit along $1/n$; monotonicity gives the full limit. □

Both hypotheses are necessary. *Infinitely many designs*: enumerate the rationals as (c_j), take targets $y \in [0,1]$ with $H(y) = y$, and let parameter-free design j have $F_j = G_j = c_j$. Exact success covers only the rationals and is null, yet density supplies an approximate witness for every target at every positive tolerance. *Noncompact parameter sets*: let $y \in [0,1]$, $H(y) = 0$, and use $(u,t) \in [0,1] \times (0,\infty)$ with $F(u,t) = u$ and $G(u,t) = 1/t$. No exact joint solution exists, yet $u = y$ with t large gives arbitrarily accurate solutions for every target.

D SUMMARY OF ASSUMPTIONS

Theorem 5 and Corollary 6 and Theorem 8 use only Definitions 1 and 3 and finiteness of $\mathcal{S}$ via Lemma 2. No smoothness, no measure theory, no saturation, and no choice of request protocol.

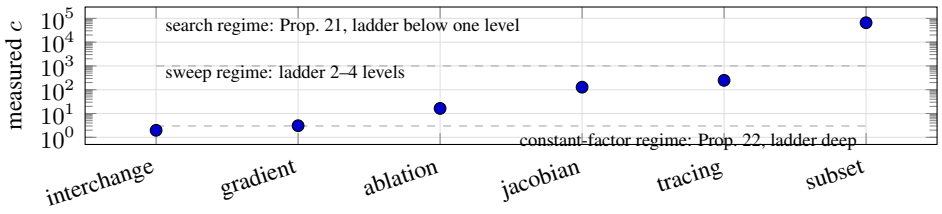

Figure 2: Measured cost ratio for the queries of Table 1, log axis, against the three predicted regimes. The gradient query lands at the constant factor Proposition 22 predicts; exhaustive search is off the scale at deployment shapes, where the pass count is 2^{4608}.

Theorem 13*(i)* uses the accounting model of Section 2, in particular unrestricted offline selection and the charging of stored constants to P, together with finiteness of $\mathcal{H}$. It uses nothing else, and it alone establishes the degeneracy. Theorem 13*(ii)* additionally requires Theorem 28, which uses C^1 response maps, a countable design set, absolute continuity of the target distribution, and the rank condition. That rank condition is the strongest assumption in the paper; it is confined to Appendix B and fixes only the tightness constant for the protocol Section 4 rejects. Both halves are statements about the pointwise protocol only.

Theorem 17 uses the uniform protocol, the charging of circuit structure to L, and the convention that presented targets are request input. Corollary 19 depends on it. Theorem 20 uses branch-free arithmetic and nothing else. Proposition 21 is conditional on $\mathrm{P} \neq \mathrm{NP}$. Proposition 22 cites Baur–Strassen under its arithmetic cost model. Proposition 29 adds compactness and a finite design class.

Theorem 25 adds the oversight premise stated at the end of Section 8, a modeling assumption and not a consequence of any theorem here.

No optimality property of any training algorithm is used anywhere.

E Measurement detail

Let M internally apply a matrix $W \in \mathbb{R}^{n\times n}$, and let its retained task capability be the action of W on presented vectors, which identifies the entries of W and gives $p = n^2$. Let the mechanism question be the signed volume scaling $H(W) = \det W$, a polynomial of degree n: a legitimate member of a broader mechanism specification, and exactly the kind of global property of an internal transformation that interventions are meant to expose. Task mode costs $2n^2 - n$ arithmetic operations, mechanism mode costs $\frac{2}{3}n^3$ by LU factorisation, and Theorem 20 gives a lower bound of $\lceil \log_2 n \rceil$ multiplications. Table 2 in Appendix E reports exact leading-order counts and median wall-clock over 25 repeats; the measured c runs from 17 at $n = 64$ to 78 at $n = 2048$.

Here every quantity is checkable in closed form, and the measured c of 17 to 78 sits far above the $\lceil \log_2 n \rceil$ bound, as expected of a bound counting multiplications alone. The determinant is not a question interpretability research asks; it is used because it is the one case where the whole theory, certificate included, is available analytically. Section 6.2 supplies the queries the field does ask.

F Further scope conditions and deployment qualifications

Proposition 30 (Decomposition does not escape the budget)**.** *Let $A_1, \ldots, A_k$ be deployments offered jointly as an interpreter of M, and let E be the ensemble: their pooled parameter vector, their circuits, and a router. Then E is an admissible deployment with $P_E \leq \sum_i P_i$, with equality when the components share no coordinates, $L_E = \sum_i L_i + O(1)$, and deadline the maximum of the components' if they are run in parallel and the sum if sequentially. Exactly one of the following holds. Either $\mathrm{Cap}(M) \not\subseteq \mathrm{Cap}(E)$, in which case E does not completely understand M under Definition 3 and Corollary 9 does not apply to it; or E retains $\mathrm{Cap}(M)$ and satisfies $\mathcal{H}$, in which case $E \succ M$ and $E \notin \mathcal{C}(\mathcal{R})$.*

Table 2: Closed-form instance. Operation counts exact to leading order; times median over 25 repeats on one CPU core. The measured ratio tracks the $\Theta(n)$ operation ratio sublinearly, the cubic routine using the memory hierarchy better than the quadratic one.

n	operation ratio	task (μs)	mechanism (μs)	measured c
64	21.5	2.0	34.1	17.3
128	42.8	2.7	70.2	25.6
256	85.5	5.8	209.1	36.3
512	170.8	21.0	1231.9	58.8
1024	341.5	155.1	6932.9	44.7
2048	682.8	566.6	44296.3	78.2

Partial understanding is untouched. The bounds above concern the contract of Definition 3 and constrain nothing weaker. Partial explanation, specialist interpreters, probing, and graded faithfulness (Jacovi & Goldberg, 2020; Adebayo et al., 2018; Hewitt & Liang, 2019) are unaffected, and by Corollary 18 a partial interpreter may select queries at the cheap end of a specification.

Offline analysis and distillation. The theorems constrain the online deployment envelope. Unbounded offline analysis, arbitrarily many queries to M, and distillation into a cheap runtime monitor are unaffected. This is not only an escape: Theorem 25 runs on it.

Weak supervision, debate, amplification. These deny the oversight premise rather than the theorems (Burns et al., 2023; Irving et al., 2018; Christiano et al., 2018). They change the available interaction or aggregate computation, and our accounting charges for the resulting system only if it is offered as a complete standalone interpreter.

Self-coverage. Horn (a) is a genuine possibility for any system with spare online capacity relative to its mechanism specification, and partial introspective ability is empirically documented (Kadavath et al., 2022; Lindsey, 2025). Theorem 20 gives one route to excluding it and Section 6 measures one instance.

Containment, not elimination. The unexplained system still exists and is more capable than anything in production. The rule transfers risk from deployment surface to containment, and its case is that fewer copies, no adversarial users, and no public interface is a large reduction rather than an elimination.

Definitional hazard. If the retained system runs as a live monitor over the deployed one, it is deployed under any reasonable reading and the rule self-destructs. The coherent version restricts it to offline analysis producing a distilled artifact that ships, and whether distillation preserves mechanism coverage is open.

Cost. Corollary 19 makes the cost a latency cost rather than a capacity cost, which is favorable: holding back a generation does not mean forgoing parameters, only accepting that the interpreting system runs slower than anything deployed. Section 6 indicates the factor is large enough to constrain ladder depth.

Numerical regularity. Theorem 5 needs none. Theorem 28 needs C^1 response maps, and piecewise systems require separate treatment of branches and boundaries. Finite precision, unbounded architecture families, and free target access must not be silently absorbed into the differentiable results.

Deployment cost is not construction cost. Our bounds constrain deployed resources. Computing a mechanism answer offline can be arbitrarily hard, and for many natural queries provably is (Barceló et al., 2020; Adolfi et al., 2025). A cheap interpreter nobody can construct is not a practical route to oversight; the two literatures bound different resources and neither subsumes the other.

G Related work in detail

Vingean reflection and inference limits. Fallenstein & Soares (2015) argue that predicting a stronger successor in full detail would already confer the predicted ability. Theorem 5 is the finite-budget, specification-relative form of that intuition, adding the dichotomy structure, the protocol analysis, and the deployment consequence. Wolpert (2008) proves inference devices cannot strongly infer themselves, using a relation that requires forcing the other device's setup rather than our retention contract. Fallenstein et al. (2015) show randomization resolves some self-referential prediction problems, illustrating that the computational model matters. Zhang et al. (2026) discuss architectural obstacles to recursive self-improvement; our proof requires no infinite sequence of metacognitive queries.

Explanation complexity. Barceló et al. (2020) and Adolfi et al. (2025) establish that specified explanation queries can be intractable. Those results bound the cost of computing an explanation; ours bound the resources of deploying one, with offline selection unrestricted. Proposition 21 is the point of contact: their hardness results supply instances where our time gap is superpolynomial.

Faithfulness and causal specification. Jacovi & Goldberg (2020) separate faithfulness from other desiderata. Adebayo et al. (2018) and Hewitt & Liang (2019) show readable explanations and predictive probes can fail to reflect the model. Geiger et al. (2021) give an operational route to specifying mechanism claims via interchange interventions, motivating our externally specified $\mathcal{H}$. Definition 15 formalizes a property their framework already assumes, namely that a mechanism claim is about a class of systems rather than one.

Models explaining and inspecting models. Bills et al. (2023) use language models to explain neurons in others, observing that larger explainers do better without establishing that every adequate explainer must be stronger. Kadavath et al. (2022) and Lindsey (2025) document partial, context-dependent self-knowledge. All three bear on horn (a), which the dichotomy accommodates. Gollakota et al. (2025) characterize when predicting a predictor's loss beats its own estimate via multicalibration failures; our $\mathcal{H}$ need not predict loss and our bounds concern realizability under a deadline.

Capacity, identifiability, and sharing. Sussmann (1992) connects input-output behavior to weights under architectural restrictions. Li et al. (2018), Aghajanyan et al. (2021), and Hinton et al. (2015) argue against inferring exhaustion from parameter count, which is why Definition 1 avoids saturation entirely. Maurer et al. (2016) and Elhage et al. (2022) show abilities need not occupy disjoint weights; we permit arbitrary sharing. Baur & Strassen (1983) and Baydin et al. (2018) give the constant-factor derivative result underlying Proposition 22. Yu et al. (2020) and Sener & Koltun (2018) study gradient conflict, weaker than absence of a joint solution in a budget. Hoffmann et al. (2022) distinguish compute-optimal training from deployment capability maximality, and Russell & Subramanian (1995) and Hartmanis & Stearns (1965) motivate declaring an architecture class and resource measure before any impossibility claim has definite content.

Executable explanations. Ghasemlou (2026) bounds the size of standalone executable explanation packages against the shortest equivalent behavioral program, motivating the full-cost accounting of $\mathcal{C}(\mathcal{R})$. Description length and stored parameter coordinates are different resources, and that bound is not used as a premise here.