

Task-local programs and the limits of bounded discovery

An inspectable execution demonstration and a finite identifiability case study

Shubham Attri · Independent researcher; founder, Phi9

0.2 — working manuscript for expert critique, not peer reviewed

Abstract

We report two small, inspectable software studies: inference-time synthesis of a task-local program followed by validation and execution in a separate bounded interpreter; and a finite Boolean discovery study that exposes the difference between a unique surviving hypothesis and an identified mechanism. A single real Codex turn produced a four-operation JSON program whose readout matched a supplied target. Fifteen targeted interpreter tests passed. In a separate frozen 1,440-run discovery matrix, recovered and replayed after reporting corrections, 518 runs produced exact singleton predictions, 272 produced incorrect singletons, and 650 produced no singleton. Under an explicitly conservative full-domain acceptance rule, all 1,440 runs abstained. Exhaustive checking establishes a finite obstruction: every three-bit cubic target agrees with some quadratic function on every seven-input subset, while observing all eight inputs costs 13 and the protocol permits only 12. The work demonstrates an execution boundary and diagnoses an experiment; it does not establish general discovery, sample-efficiency superiority or a new physical law. A subsequent offline integration executed and replayed 5,856 learner cases. Its separate exhaustive factorial control recovers all 128 cubic truth tables under three policies only when both a complete hypothesis family and sufficient observations are supplied. The same control exposes misspecification under a restricted family. This does not establish equal-spend superiority.

Phi9 thesis and identity

Phi9 is an independent research initiative asking how measurements, representations and dynamics constrain learning. Our thesis is a question-led program: establish what information a task needs; build an inspectable computation; test its assumptions against matched controls; and preserve the limits of the result. The general physics-of-learning ambition is a research direction, not a proven theory. This paper is one evidence contribution to that story. A proposed first collaboration is a scoped trace-and-assumption review of a synthetic or explicitly permitted example, with an annotated evidence map and next-test recommendation. No customers, commercial readiness or validated demand are established.

What is being contributed

The practical output is an auditable, restricted program interface and a negative case study with corrected resource admission, acceptance reporting and reproducibility checks. The mathematical argument is elementary finite Boolean algebra, not a claim of novelty. The research question is whether future systems can distinguish search limitations, hypothesis-class limitations and observational insufficiency without privileged state access.

Evidence boundary

The two retained studies are now connected by a demonstrated offline research workflow. Its final current-code run executed 5,856 learner cases and deterministically replayed them, alongside archived-program execution, a representation workload and versioned artifact generation. Seventy-five engineering tests passed. No fresh model inference, model-weight updates, global permission changes, live sensing, autonomous repair or hosted deployment are demonstrated. Original frozen and new extension outcomes are reported separately.

2. Model, representation and inquiry

Task-relative information

Let x be a system state, $r(x)$ a representation, F_a a permitted transition and q the task readout. Exact closure for a task requires a represented transition T_a and readout h such that $r(F_a(x)) = T_a(r(x))$ and $q(x) = h(r(x))$ for the states and interventions in scope. If two states have equal representations but different required readouts or represented successors, no deterministic function of that representation alone can be exact. This is a restricted sufficiency condition, not a universal account of learning or physics.

Finite identifiability

For hypothesis family H and observed history $D = \{(u_i, y_i)\}$, define $V_H(D) = \{f \text{ in } H: f(u_i) = y_i \text{ for all } i\}$. A singleton $V_H(D)$ identifies a function only conditional on H containing the true mechanism. An empty version space can signal misspecification, but a nonempty version space cannot rule it out. To justify a stronger claim, one needs adequate observations and justified assumptions about the target family. Evaluator truth labels are not legitimate extra observations for a learner.

Geometry and programs

The interpreter represents simulated state as a finite numerical vector and an optional scalar. Its geometry is an encoding with specified arithmetic transformations; coordinates do not by themselves establish physical meaning, causal validity or a learned manifold. Program tokens serialize a plan. Arithmetic occurs in a separate interpreter. Task-local plan synthesis must not be described as changing model weights or granting tools new permissions.

Separating the two resource limits

The new factorial control holds the observation and hypothesis factors apart: degree-two versus complete degree-three families at cost caps 12 and 13. Early singleton stopping is disabled identically within that control. It demonstrates a finite separation, not a general claim that more search or more sensing alone always suffices. The primary comparison retains the original stop rule and is analyzed separately.

3. Study A — inference-time program synthesis

Procedure

One actual Codex CLI 0.159.3 generation turn, bounded to at most 120 seconds, received a restricted DSL specification, synthetic initial vector $[1,2]$ and a task with its expected answer. The returned JSON was parsed, validated and executed by a separate Python interpreter. The archived generated program and receipts support replay without another model call. Supplying the expected answer makes this a plan-synthesis demonstration, not independent discovery.

Executed transformation

$[1,2] \rightarrow \text{scale}(2) \rightarrow [2,4] \rightarrow \text{add}([1,-1]) \rightarrow [3,3] \rightarrow \text{dot}([1,1]) \rightarrow \text{scalar } 6 \rightarrow \text{readout}$. The trace reports four instructions and nine abstract compute units. Deterministic replay and command-line replay matched the stored domain outputs. Independent task evaluation returned true. Fifteen targeted tests passed; these are engineering checks, not fifteen independent model trials.

Admission contract

Only scale, add, dot and readout are allowed, with exact instruction keys. Raw JSON is capped at 32,768 UTF-8 bytes; duplicate keys, unsupported fields and nonfinite numbers reject. Host maxima are 64 instructions, 4,096 abstract compute units, 17 logical state elements and 65 output records. Dimensions are 1–16 and numerical magnitude is bounded. A program can narrow these caps but cannot increase them. Each operation reserves instruction, compute, state and output capacity before arithmetic. A valid program can abstain when admission fails; an admitted arithmetic failure preserves the last valid state.

What the limits mean

Compute charges are coordinate-operation accounting, not measured CPU cycles or wall-clock limits. Logical state bounds do not measure total process memory. Output records are not model tokens or bytes. Provider token and monetary usage totals were not captured. The DSL has no filesystem, network, shell, import or tool opcodes, but its trusted Python host is not a general sandbox. Empty programs can complete without a readout, so task success must be checked separately from sequence completion. A scalar is the latest dot result; subsequent scale/add operations do not recompute it.

4. Study B — protocol and finite obstruction

Task and comparison matrix

The task is noiseless membership-query discovery of Boolean functions on three bits. Targets are affine, genuinely quadratic, genuinely cubic, or a restricted-input condition. Three policies (random, maximum current-survivor output entropy, entropy divided by probe cost) are compared under fixed versus expandable repertoires. Sixty evaluation seeds per cell give $4 \times 3 \times 2 \times 60 = 1,440$ runs. Fixed models begin at degree one; expandable models can reach degree three, but expansion occurs only when the version space becomes empty.

Resource and stop rules

At most eight observations, observation cost 12, 400 generated candidates and 50,000 abstract learner-work units are admitted. Probe cost is $c(u)=1+\max(0,\text{popcount}(u)-1)$. A singleton receives two audit probes before the original early-stop condition can fire. These probes are not an equivalence oracle and cannot prove global correctness. The recovery introduces full-domain acceptance only when a singleton is consistent with observations at all eight inputs. This intentionally conservative rule is impossible at cap 12. Equal ceilings are not equal actual expenditure.

Elementary obstruction

Work over $\text{GF}(2)$. Every function on three bits has a unique algebraic normal form using constant, linear, pairwise and triple monomials. For any omitted input v , its point indicator $\delta_v(u)$ is the product over three coordinates of u_i when $v_i=1$ and $(1+u_i)$ when $v_i=0$. It is zero except at v and has coefficient one on the triple monomial. For any genuinely cubic target f , $g=f+\delta_v$ cancels that coefficient and has degree at most two. Therefore g agrees with f at the other seven inputs but differs at v . No seven-input history can distinguish the two without additional assumptions.

Cost and confirmation scope

The costs for the eight inputs sum to $1+3\times 1+3\times 2+3=13$. The cap is 12. Exhaustive retained checking evaluated $128 \text{ cubic targets} \times 8 \text{ omitted inputs} = 1,024$ target/subset pairs and found a quadratic match in every pair. This confirms the finite algebraic obstruction; it is not an impossibility theorem for arbitrary noisy, continuous or physical tasks.

5. Results and corrected interpretation

Aggregate outcomes

Across the recovered 1,440-run matrix: 518 exact singleton predictions (36.0% of runs); 272 incorrect singleton predictions (18.9%); 650 nonsingleton/no-model outcomes (45.1%). Among the 790 singleton outcomes, 272 were incorrect (34.4%). These are descriptive pooled counts across deliberately different conditions, policies and repertoires, not a deployment success rate. Conservative accepted answers: 0; abstentions: 1,440.

Cubic and affine cases

All 180 expandable cubic runs ended with incorrect singleton predictions. Because a quadratic match remains for each seven-input subset, expanding only on an empty version space cannot reliably expose cubic misspecification. The 360 affine runs produced exact singleton predictions, but still did not satisfy full-domain acceptance. Reporting both facts prevents evaluator success from being confused with learner certification.

Revision versus original

Recovery retained the original files and replayed all raw runs. Fourteen original mechanical tests passed. A separate revision added pre-generation budget admission, separated singleton/audit/acceptance/abstention fields, renamed the adaptive-unobserved error and checked matrix completeness and duplicate keys. Eighteen revised tests passed and all revised runs replayed. Standard-budget trajectories and survivors were unchanged. This is a reporting and admission correction on the same seeds, not an independent confirmation sample or improved discovery performance.

Measurement caveats

The unobserved error is evaluated only for singleton predictions and only on inputs not selected by an adaptive learner; undefined cases are null, not zero. It is not error on a fixed independent held-out set. Learner-work units and evaluator-work units are distinct. Separate deterministic random streams are not a proof of statistical independence. Protocol provenance mentions earlier exploratory attempts not retained in the recovered directory; those are not reconstructed or silently treated as preregistered evidence.

Interpretation

The useful result is a precise failure mode: uniqueness inside a misspecified hypothesis family can masquerade as certainty, and additional search cannot recover a distinction that the allowed observations do not resolve. The evidence does not show general sample-efficiency superiority. The completed, separately protocol-pinned factorial extension is reported next. Its results do not overwrite these original 518/272/650 counts.

6. Integrated system and controlled extension

What the final workflow executed

The final current-code offline run connects vendored byte-pinned components: 1,440 fresh frozen-protocol baseline cases, 2,880 new matched primary cases and 1,536 exhaustive controls; 5,856 learner cases in total. It also executes the archived Codex JSON through the real interpreter, calls the existing research-core calculation kernel, and builds pinned versioned artifacts through the retained pipeline. Deterministic replay re-executes the workloads and compares domain outputs, excluding timing samples. Seventy-five tests passed (9 system, 15 token, 18 discovery, 33 artifact). These are engineering checks, not 75 model trials. The extracted package replay also verified. Initial integration/replay failures remain retained in attempt history; only runs/ready-v1 validates the packaged current code.

Matched primary comparison — new seeds, original stopping

A distinct 30000–30059 seed block retains the original two-singleton-audit stopping rule and changes only the observation cap. Cap 12: 514 exact singletons, 273 wrong, 653 other outcomes, 0 accepted and 1,440 abstentions. Cap 13: 653 exact, 134 wrong, 653 other, 293 accepted and 1,147 abstentions. Each arm has 1,440 cases. These new counts are not the original frozen 518/272/650 counts. Actual observation-cost totals increase from 12,035 to 12,605; modeled learner-work totals increase from 915,702 to 1,485,533. The cap increase does not eliminate early stopping or repertoire limitations.

Exhaustive factorial — a separate stop policy

All 128 cubic truth tables are enumerated under three policies, giving 384 cases per grammar/cap cell. Both caps use the same no-early-singleton stop rule; initial degree and observations are varied explicitly. Quadratic-only family at cap 12: 384 wrong singletons, all abstain. Quadratic-only at cap 13: 384 no-model outcomes, all abstain. Complete degree-three family at cap 12: 384 ambiguous outcomes, all abstain. Complete degree-three at cap 13: 384 exact singletons, all accepted. Within this finite construction, adequate observations expose a restricted family as inadequate; a complete family without the discriminating observation remains ambiguous. This control differs from the primary stop rule and must not be pooled as if only the cap changed.

Costs, representation and limitations

Factorial actual observation spend is 10 per case at cap 12 versus 13 at cap 13. The complete family doubles the candidate inventory from 128 to 256 and increases modeled computation. Hence the result is not equal-spend efficiency superiority. The separate 64-integer representation kernel preserves all 72 queries with 256-byte raw and 86-byte lossless payloads; 64-byte bin means preserve eight aligned sums but none of 64 point queries. Those payload sizes are not process memory, and one timing sample is not a speed ranking. No noisy, held-out physical or learned-representation benchmark is supplied. The offline integration is a functioning reproducible workflow, not a hosted autonomous scientist or commercial-ready service.

7. Related work and scope

Abstraction and task-dependent memory [1-3]

Abel et al. study near-optimal abstraction transfer across a specified distribution of RL tasks [1] and abstraction as a compression-performance tradeoff in apprenticeship learning [2]. These are relevant precedents, not methods that ignore change. Booker and Majumdar jointly learn task-centric memory and control policies with group-LASSO regularization [3]. Our finite case study does not challenge their stated guarantees or evaluate their algorithms. It isolates a separate question: which observational distinctions are required for a claim after the model family changes?

Generated programs and library learning [4-6]

PAL [4] and Program of Thoughts [5] already separate model-generated program reasoning from external computation. Our JSON interpreter is a restricted, inspected implementation of that broad boundary, not its invention. DreamCoder [6] learns reusable program libraries and a search policy through wake-sleep synthesis; our prototype has neither learned library acquisition nor that training loop. One successful known-answer program does not establish synthesis generalization or efficiency.

Video representations and procedural tasks [7-8]

V-JEPA learns video representations through feature prediction [7]. It motivates downstream evaluation rather than equating latent reconstruction error with task preservation, but no V-JEPA downstream experiment is reported here. EgoPER procedural-error detection [8] is relevant to a possible future permitted packaging observation pilot; transfer to packaging is untested. A noncommercial toy protocol, consent and session-level holdouts would be prerequisites, not evidence of a commercial dataset or buyer.

Limits and next falsifiable test

The current environment remains finite, noiseless and synthetic. The completed factorial illustrates why both observation access and a sufficient hypothesis family matter; the original-stop primary arm shows that merely raising a cap does not force a learner to use it. Next tests need noisy or physical tasks, matched actual costs, explicit stop policies and independent task evaluation. No such generalization has been measured.

8. Reproducibility and references

Retained executable evidence

Study A: phi9-token-program-v1 contains SPEC.md, interpreter.py, codex-program.json, execution-trace.json, evaluation.json, tests.txt and generation receipts. Run: `python3 interpreter.py < codex-program.json; python3 -m unittest -v test_interpreter`. Study B: the recovered package contains frozen baseline and revision, protocol.json, freeze.json, results.jsonl, summary.json, identifiability-check.json, tests and recovery receipts. From the revision directory: `python3 -m unittest -v; python3 discovery.py audit; python3 recover_evidence.py`. Do not run `prepare.py` as a replay or overwrite frozen files. See the companion reproducibility guide for durable attachment IDs and SHA256 values obtained from the retained files. The integrated POSIX/Linux Python-standard-library release runs with `python3 release.py run --output runs/my-run`, then `python3 release.py replay --output runs/my-run`. Use a new output path; run refuses existing directories. The delivered package replays `runs/ready-v1`. Status reads receipts rather than revalidating current files. See the companion guide for the retained final-run evidence and commands.

References

- [1] Abel, D., Arumugam, D., Lehnert, L., Littman, M. L. (2018). State Abstractions for Lifelong Reinforcement Learning. ICML, PMLR 80. <https://proceedings.mlr.press/v80/abel18a.html>
- [2] Abel, D., Arumugam, D., Asadi, K., Jinnai, Y., Littman, M. L., Wong, L. L. S. (2019). State Abstraction as Compression in Apprenticeship Learning. AAAI. <https://aaai.org/papers/03134-state-abstraction-as-compression-in-apprenticeship-learning/>
- [3] Booker, M., Majumdar, A. (2021). Learning to Actively Reduce Memory Requirements for Robot Control Tasks. L4DC, PMLR 144:125–137. <https://proceedings.mlr.press/v144/booker21a.html>
- [4] Gao, L. et al. (2023). PAL: Program-aided Language Models. ICML, PMLR 202:10764–10799. <https://proceedings.mlr.press/v202/gao23f.html>
- [5] Chen, W., Ma, X., Wang, X., Cohen, W. W. (2023). Program of Thoughts Prompting: Disentangling Computation from Reasoning for Numerical Reasoning Tasks. TMLR. <https://openreview.net/forum?id=YfZ4ZPt8zd>
- [6] Ellis, K. et al. (2021). DreamCoder: Bootstrapping Inductive Program Synthesis with Wake-Sleep Library Learning. PLDI. <https://doi.org/10.1145/3453483.3454080>
- [7] Bardes, A. et al. (2024). Revisiting Feature Prediction for Learning Visual Representations from Video. TMLR. <https://openreview.net/forum?id=QaCCuDfBk2>
- [8] Lee, S.-P., Lu, Z., Zhang, Z., Hoai, M., Elhamifar, E. (2024). Error Detection in Egocentric Procedural Task Videos. CVPR. https://openaccess.thecvf.com/content/CVPR2024/html/Lee_Error_Detection_in_Egocentric_Procedural_Task_Videos_CVPR_2024_paper.html

Availability and declarations

This is an independently authored Phi9 working manuscript for critique. No institutional endorsement, peer review, buyer demand or production readiness is claimed. Codex generated the archived restricted program; assistant tooling prepared documentation and packaging. No publication, outreach send or external submission occurred in preparing this edition. Correspondence and research-provenance working notes are excluded from the public media package. Existing phi9.space is an identity link, not proof that this new edition has been deployed.