Beyond the Original Problem

Transformational Intelligence for Expanding What Is Possible

Vincent Boucher
MONTREAL.AI
Montréal, Québec, Canada

October 19, 2026 · Research manuscript · Revised edition

Abstract

An AI can become extraordinarily capable at solving a problem whose formulation is the principal obstacle to progress. We study a different target: *recognize when the original problem formulation is unnecessarily restrictive, discover a fundamentally different game to play, and demonstrate that the change better serves the purpose that justified the work*. The object of invention is not merely an answer, strategy, or reusable tool. It is the problem-solving setup: what must be represented, what information is acquired, which intermediate goals are necessary, how work is coordinated, and what may replace the original task. We formalize purpose-relative attainable outcomes and distinguish a better solution, a useful reformulation, and a demonstrated escape from a formulation-dependent bound. A new exact finite study isolates complementary changes: reconstructing an eight-bit world requires eight binary probes; changing only the measurement protocol or only the intermediate objective still requires eight; changing both permits all seven required answers to be obtained from three probes. At a three-probe budget, optimal coordinate-only investigation has full-answer success $1/8$, whereas purpose-directed measurement achieves 1 under the specified model. A finite constructor discovers an adequate design, a separate checker validates every required output, and an ordinary unrestricted linear-algebra baseline matches it. These are synthetic, classical, model-conditional results, not a novel sensing algorithm or an LLM breakthrough. A retained dynamic-state study shows why current correctness does not justify forgetting future obligations. Together, the results motivate an architecture and evaluation protocol for discovering better formulations, including combinations that isolated optimization misses, tests that create decisive evidence, and transformations that should be rejected. Eliminating unnecessary work is one possible consequence. The broader objective is to expand what can be achieved.

Keywords: transformational intelligence; problem reformulation; purpose-relative capability; experiment design; persistent capabilities; discovery evaluation.

Transformational intelligence recognizes when the original problem formulation is unnecessarily restrictive—and discovers a fundamentally different game to play.

The purpose is the reference. The formulation is a candidate. The improvement must be demonstrated.

1 The original formulation is a hypothesis, not a law of nature

Suppose an AI is asked to summarize a book. A second AI searches the source for consequential omissions, and a third checks the criticisms. That is a useful change of strategy. But it leaves a deeper question untouched: why should understanding begin with a compressed artifact that must later be audited for what it lost?

A source-grounded argument map might make a required summary more faithful. When the actual authorized goal is preparation for a discussion, an adaptive evidence interface might serve it better than a fixed summary. When the required artifact really is a two-page summary, replacing it with an interface is not an improvement merely because the AI prefers the interface. The creative opportunity lies in separating the intended outcome from an inherited proposal for producing it.

The same distinction appears in mathematics, software, research, and coordination. A mathematical investigator may replace instance-by-instance search with a general construction. A scientific agent may stop elaborating explanations and invent the experiment that distinguishes them. A computer-use agent may discover a permitted way to remove an unnecessary interaction sequence. A coordinator may remove a dependency instead of scheduling more workers around it. Each change must be tested; none is justified by novelty alone.

The phrase "different game" does not require an opponent. It means a changed decision structure: different representations, evidence, dependencies, intermediate targets, or ways of producing the relevant outcome. It is not permission to change physical facts, evade an obligation, hide costs, or substitute the system's objective for the user's.

Our earlier *Transformational Intelligence* manuscript made formulation change the object of discovery (Boucher, 2026). The first edition of the present work, *Beyond Task Completion*, emphasized persistent capabilities that make recurring work unnecessary. That is an important special case, but an unnecessarily restrictive thesis for the broader project. A transformation may instead make a previously infeasible outcome attainable, acquire a decisive observation, enable a new kind of construction, or reveal a better question that requires explicit authorization. It may temporarily require *more* work. Minimizing work is not the definition of intelligence.

1.1 The question this paper asks

The conventional optimization problem is: *given a formulation, find a better policy*. The problem investigated here is: *given an authorized purpose, determine whether the formulation itself is a binding restriction, construct an alternative, and establish whether changing it was worthwhile*. More agents, better prompts, or larger models may help either problem, but do not distinguish them.

The decisive question is therefore not "Did the agent produce a clever reframing?" It is:

> Did the system identify a genuinely removable restriction, realize a different setup, preserve the relevant obligations, and achieve something that the appropriate comparison could not achieve as well under the declared resources?

A fluent explanation answers none of these questions by itself. Nor does a long checklist of controls invent the alternative. Discovery and demonstration are complementary tasks.

1.2 Contributions and evidential scope

This revision develops a purpose-relative account of attainable outcomes; an exact joint-reformulation study with a complete restricted-frontier calculation; a discovery architecture centered on diagnosed restrictions and executable alternatives; and an evaluation protocol that tests discovery separately from supplied engineering. The previous finite-controller study is retained as a test of future validity, not as the entire definition of

transformational intelligence.

The conceptual roots are substantial. Boden distinguished exploring a conceptual space from transforming it, and Ritchie examined what would make this distinction precise and empirically testable (Boden, 1998; Ritchie, 2006). Automated abstraction, experimental design, program synthesis, and agent design offer direct technical precedents. We do not claim to originate these fields or provide a universally superior discovery algorithm. The contribution is a testable operational synthesis, explicit formal distinctions, and executed reference constructions that expose both the opportunity and the traps.

2 What changes when an AI changes the game?

2.1 Keep four objects separate

An assignment can entangle four different objects. The *purpose contract* states the desired outcomes, material constraints, authority, and resource limits. The *formulation* specifies one way of representing and pursuing that purpose. The *policy* chooses actions within that setup. The *evaluation instrument* estimates whether the actual outcome is good. Improvement becomes ambiguous when these objects are conflated.

For a fixed comparison, write

$$\mathcal{I} = (\mathcal{Z}, O, \mathcal{H}, \mathcal{A}, \mathbf{B}, H), \tag{1}$$

where $\mathcal{Z}$ describes initial conditions and exogenous events, O identifies outcome variables and their preference ordering, $\mathcal{H}$ states hard requirements, $\mathcal{A}$ states external permissions, $\mathbf{B}$ is a resource vector, and H is the relevant horizon. Expected-value comparisons fix a declared law for Z on $\mathcal{Z}$; worst-case comparisons quantify over the specified admissible conditions. A purpose can be clarified or amended. Material amendments require the appropriate authority and start a new comparison; they do not retroactively improve an old score.

A formulation can be represented as

$$F = (\mathcal{R}, Q, \mathcal{D}, \Pi, \nu), \tag{2}$$

with representation $\mathcal{R}$, information protocol Q, work and coordination dependencies $\mathcal{D}$, executable policy language Π, and intermediate objective ν. These are not independent in general. A useful information protocol may require a new representation; a different representation may require a compatible solver. The new study makes this interaction explicit.

Definition 2.1 (Transformational change)**.** *Relative to a named incumbent formulation, a transformational change replaces a restriction in the problem-solving setup rather than merely improving the policy inside it. A* demonstrated useful transformation *additionally realizes the change, respects the purpose contract, and supports a relevant improvement after the declared costs are included.*

A genuine transformation can fail. A valuable improvement can be nontransformational. The first classification describes *what changed*; the second concerns *whether it helped*. Neither is established by the size of an edit or the sophistication of its description.

2.2 Five possible changes of game

Evaluation and discovery procedures can also contain restrictive assumptions, but their revision has distinct evidential obligations. A candidate cannot acquire authority by writing a new contract, and cannot validate itself by choosing a more accommodating judge. Conversely, permanently freezing an inadequate evaluator can itself suppress useful transformations.

Table 1: Mechanisms to investigate, not a complete catalogue or automatic entitlement to change the task.

Object changed	Inherited restriction	Candidate alternative
Representation	Every source distinction must remain explicit	Find an invariant, sufficient state, factorization, or new language
Information	Progress must use the evidence already in hand	Invent a permitted experiment or change what is measured
Dependencies	Current handoffs and serial steps are necessary	Construct an interface or shared object that removes avoidable dependencies
Intermediate objective	The whole world or original artifact must be reconstructed	Produce exactly what the authorized outcome requires through a different construction
Task production	Each recurring request requires another full investigation	Build a conditional capability that changes future work

2.3 The right invariant is not the incumbent's complete trace

Demanding that a new system reproduce every internal task, observation, or message can forbid the very change under investigation. We instead compare the aspects of behavior that the purpose actually requires. Let $\Gamma_{\mathcal{I}}(\tau)$ project a full execution trajectory τ onto relevant outcomes, including required timing, side effects, and future obligations. A transformation can change τ while preserving or improving $\Gamma_{\mathcal{I}}(\tau)$.

The projection is not chosen unilaterally by the candidate. Omitting a required action from $\Gamma_{\mathcal{I}}$ is an invalid contract, not a breakthrough. Some domains require exact output traces; others permit different artifacts that serve a common outcome. The appropriate bridge may be a mathematical proof, an independent outcome checker, a controlled experiment, or a validated measurement instrument. These are different forms of evidence with different limits.

Proposition 2.1 (Internal task counts are not an outcome invariant)**.** *When internal labels are not independently prescribed obligations, splitting one execution into multiple labeled tasks or fusing adjacent labels can change the number of completed tasks without changing the execution's required outcomes.*

Proof. Partition an internal sequence into m nonempty segments, or erase the boundaries between adjacent segments, without altering actions or required external behavior. The label count changes; the outcome projection does not. □

Task counts can remain diagnostic. They cannot, alone, establish the value of a transformation. The same warning applies to agent counts, tokens saved, or the number of ideas generated.

3 From a better answer to a larger attainable set

3.1 Purpose-relative capability

Let $\mathrm{Run}(F, \pi; z)$ include setup and execution under exogenous conditions z. Suppose numerical outcomes $g_{\mathcal{I}}$ are oriented so that larger is better; hard requirements are assessed separately. For a declared resource boundary, define the downward-closed attainable set

$$C_{\mathcal{I}}(F, \mathbf{B}) = \left\{ v : \begin{array}{l} \text{some admissible } \pi \text{ satisfies the hard requirements and resource limits,} \\ \mathbb{E}[g_{\mathcal{I}}(\mathrm{Run}(F, \pi; Z))] \succeq v \end{array} \right\}. \tag{3}$$

For an exact worst-case contract, replace the expectation by the stipulated pointwise requirement. Do not exchange a worst-case guarantee for an average score without saying so. Multiple objectives need not be

collapsed into a single number.

A realized transformation T has a new formulation F_T, construction and discovery costs $\mathbf{K}_T$, a realization map, an outcome bridge, and a statement of the restriction changed. Its relevant operating set is $C_{\mathcal{I}}(F_T, \mathbf{B}-\mathbf{K}_T)$ only when additive accounting is justified and all remaining lifecycle costs are inside the run. Nonadditive resources or interactions require their actual accounting rule.

Three claims should remain distinct. An *observed gain* beats named implementations. A *formulation-dependent gain* has evidence tying the improvement to a structural change. A *certified expansion* supplies an outcome vector attainable by the new setup but outside a valid upper bound for the old setup at the specified resources. A successful experiment against several baselines does not establish the old setup's universal limit.

3.2 Changing the game does not change computability

A sufficiently general program can encode both the old and new methods. We make no claim that a change of formulation creates a new computable function beyond what a universal machine can express. The distinction concerns which methods are expressible, discoverable, verifiable, or executable under an actual representation and resource budget.

Classical abstraction work already shows why problem representations matter for search (Knoblock, 1994). DreamCoder makes languages and reusable abstractions objects of learning (Ellis et al., 2020). In our framing, an unrestricted agent is an essential comparator precisely because it may discover the same transformation without a specially named architecture.

Proposition 3.1 (Policy transfer gives a conditional inclusion)**.** *Suppose every admissible policy in F_0 has an admissible realization in F_1 that preserves the relevant outcomes and uses no more than the same resource vector. Then $C_{\mathcal{I}}(F_0, \mathbf{B}) \subseteq C_{\mathcal{I}}(F_1, \mathbf{B})$. If constructing F_1 has additional cost, the same-budget inclusion does not follow unless that cost is covered by the transfer bound.*

Proof. Transfer the policy witnessing each member of the first set. Its preserved outcomes and resource bound witness membership in the second set. An uncharged construction cost would invalidate that witness at the original budget. □

This is a transparent implication, not an estimator of the attainable set or a practical algorithm for finding the transfer. A transformed system can improve one region of a frontier while worsening another. "A better game" is purpose- and resource-relative, not a universal ordering of all formulations.

3.3 The information needed for an outcome can be less than the information needed for a world model

More information can support more decisions, but acquiring it has a cost. The theory of comparing experiments provides a long-established decision-theoretic foundation (Blackwell, 1953); experimental design studies how to select informative observations under computational and practical constraints (Rainforth et al., 2024). Our next study isolates a different emphasis: an unnecessarily strong *intermediate objective* can make even an excellent experiment selector solve the wrong information problem.

An investigator asked to identify an entire hidden state may overlook a cheaper protocol that answers every authorized question without identifying that state. This is not permission to ignore a demanded explanation or audit record. It is a hypothesis that reconstructing the whole state was an inherited means rather than an obligation.

4 An exact study: answer every required question without identifying the world

4.1 The purpose and the available interface

A hidden world $\theta \in \mathbb{F}_2^d$ is chosen uniformly at the start of an episode and remains fixed. Arithmetic is over the two-element field. A permitted probe with mask $a \in \mathbb{F}_2^d$ returns the single noiseless bit

$$y_a = a^\top \theta \pmod 2. \tag{4}$$

The purpose is to answer every question in a specified family $Q \subseteq \mathbb{F}_2^d$, where question q requires $q^\top \theta$. No requirement asks the investigator to reconstruct θ itself. All compared systems have the same stated probe permissions from the start. A coordinate-only formulation restricts itself to masks e_i; a programmable formulation can also use the permitted noncoordinate masks.

The interface is part of the *supplied synthetic environment*, not a newly available physical oracle credited to the candidate. A one-bit programmable measurement may be more expensive than a coordinate measurement in an actual application. We first report a probe-count frontier and then vary probe prices. We do not equate a probe with a CPU operation, a dollar, or a second.

Let $U = \operatorname{span}(Q)$ and $r = \dim U$. The required output vector is the collection of all requested parities. It has 2^r possible values under the uniform prior. Only a purpose-sufficient measurement vector is needed to determine it.

Theorem 4.1 (Exact information requirement for the target family)**.** *If all masks in a basis of U are permitted, r binary probes suffice to answer every question in Q for every world. Any procedure using at most b binary probe responses, including adaptive procedures with independent internal randomness and no other world-dependent information, has full-output-vector success at most*

$$\min\{1, 2^{b-r}\}. \tag{5}$$

In particular, exact success for all worlds requires at least r binary probes. A deterministic retained answerer with no further world access needs at least 2^r distinguishable operational states if it must subsequently answer every query in Q.

Proof. Probe a basis $a_1, \dots, a_r$ of U. Write each $q = \sum_j c_j a_j$ and answer $q^\top \theta = \sum_j c_j y_{a_j}$. For the lower bound, fixing the independent randomness yields a binary decision tree of depth at most b, with at most 2^b terminal transcripts. Each transcript can correctly name at most one required output vector, whose preimage contains 2^{d-r} worlds. Thus at most $2^b 2^{d-r}$ of the 2^d equally likely worlds can receive a correct full vector. Averaging over independent randomness preserves the bound. Finally, two different required output vectors must remain distinguishable to a retained answerer: at least one permitted future query separates them. □

The mathematics is elementary linear algebra and decision-tree counting. It is not proposed as a new sensing theorem. Its role is to identify exactly which restriction must change and which performance claim can be proved.

4.2 The eight-bit instance

The main instance has $d = 8$ and

$$U = \operatorname{span}\{11111111, 01010101, 00110011\}, \qquad Q = U \setminus \{0\}. \tag{6}$$

The seven query masks, written as integers for the executable artifact, are

$$Q = (51, 85, 102, 153, 170, 204, 255). \tag{7}$$

The probe catalogue is the union of these seven masks and the eight coordinate masks: fifteen permitted operations. The search constructor receives the finite outcome and probe semantics, not an annotation naming the preferred basis. The world used in a later episode is not supplied to it.

The incumbent formulation says: read coordinates, identify the full world, then answer the questions. It needs eight probes. Improving its internal reasoning cannot identify an unobserved independent bit. Importantly, both of the following apparently substantial improvements also need eight probes for exact success.

Change only the intermediate objective. Ask for the required answers rather than the full world, but retain coordinate-only sensing. The family includes parity of all eight coordinates. If any coordinate remains unread, flipping it preserves every coordinate observation and changes that required answer. Exact success for every world still needs all eight coordinate probes.

Change only the measurement protocol. Permit programmable binary sensing, but retain the intermediate requirement to identify the complete eight-bit world. There are 256 possible worlds. Fewer than eight binary responses cannot distinguish them all, even adaptively. Exact full-world reconstruction still needs eight probes.

Change both. Acquire a purpose-sufficient measurement vector rather than a complete coordinate description. Three independent masks from U determine all seven answers. Neither a new external permission nor a weaker required output has been introduced.

Table 2: Exact minimum probe counts in the supplied instance. Each single change leaves the minimum at eight; the joint change lowers it to three. Minima follow from the stated bounds and are reproduced by a strong algebraic control over the supplied catalogue.

Intermediate objective	Coordinate sensing	Programmable sensing
Identify the complete world	8	8
Answer the required family	8	**3**

This is a concrete complementarity, not a rhetorical appeal to "thinking differently." Greedy selection that requires each isolated formulation edit to improve the probe count can stop at the incumbent. With positive edit costs, each single edit can even look worse. That does not justify unbounded exploration; it motivates testing a bounded set of mechanism-based combinations.

The useful question was not "How do we identify the world faster?" It was "Why must we identify the world to answer the questions we are actually required to answer?"

4.3 A finite constructor discovers an executable protocol

The constructor enumerates subsets of the permitted probe catalogue in increasing cardinality. For each subset P, it groups all 256 possible worlds by their probe-response vector and asks whether every group has a unique required output vector. If so, that grouping supplies a decoder. The search uses the finite table semantics; it does not call a rank routine or receive the semantic labels in Equation (6).

In the main ordering it evaluates 502 candidate probe sets and returns

$$P = (51, 85, 153). \tag{8}$$

The three-bit response has eight reachable values. A decoder with eight rows returns the required seven-bit answer vector. The saved JSON artifact contains the selected probes, decoder, contract binding, and scoped search metadata. It is data interpreted by a bounded local checker, not a package of self-authorizing executable instructions.

The search evaluates $502 \times 256 = 128{,}512$ candidate–world combinations before returning. That is a real construction ledger, but not a complete cost model: computing signatures, serializing artifacts, allocating objects, and verifying results add work. The three-probe result concerns *operating information acquisition after construction*; it is not a claim that the entire discovery process costs three operations.

A separate checker takes the expected contract as a separate input. It recomputes probe bits and required outputs through its own implementation, checks all $256 \times 7 = 1{,}792$ outcome equalities, and verifies the decoder's reachable domain. It does not invoke the constructor, its rank calculation, or its scoring routine. It does not certify that search metadata honestly describes historical execution or prove optimality; those claims have their own log and mathematical argument.

An ordinary linear-algebra baseline supplied with the same query and probe semantics also finds a three-probe design. This equality is essential. The result establishes a structural opportunity and a finite construction of it, *not superiority over unrestricted experimental design or evidence of general AI creativity*.

4.4 An exact frontier, not a weak-baseline comparison

At a budget of three probes, the transformed design returns every required answer correctly. What can the best coordinate-only procedure do under the same uniform-world model?

For a coordinate set S, let $Q_{\overline{S}}$ be the query matrix restricted to unread coordinates. Conditional on the observed coordinates, the residual output uncertainty is uniform over an affine set with $2^{\operatorname{rank}(Q_{\overline{S}})}$ values. Thus the optimal probability of returning the *entire* answer vector correctly is

$$A_{\text{coord}}(b) = \max_{|S| \le b} 2^{-\operatorname{rank}(Q_{\overline{S}})}. \tag{9}$$

Adaptive coordinate choice cannot improve this bound in this uniform linear instance: at every terminal transcript the unread bits remain uniform, and its observed set has size at most b. A fixed set attaining the maximum achieves the bound. Appendix A gives the details.

Table 3: Optimal full-answer-vector success versus probe budget in the noiseless finite model. The coordinate column is exhaustively calculated; the unrestricted binary column is the tight bound in Theorem 4.1. This is not an LLM accuracy benchmark.

Maximum binary probes	Best coordinate-only protocol	Best permitted purpose-directed protocol
0	1/8	1/8
1	1/8	1/4
2	1/8	1/2
3	1/8	1
4	1/4	1
5	1/4	1
6	1/2	1
7	1/2	1
8	1	1

The gap at three probes is an actual old-formulation bound, not merely the failure of a weak implementation. Additional computation or cooperating critics cannot remove the residual information uncertainty while the

coordinate-only restriction and probe budget remain fixed. A fully informed unrestricted policy, however, matches the new design. The limitation belongs to the named formulation, not to all possible "ordinary AI."

4.5 Negative controls: less information is not always enough

A two-probe shortcut is insufficient. For the selected design's first two masks, worlds 3 and 4 give the same observed response but different required output vectors. The saved witness records both worlds and their outputs. More confidence in the insufficient transcript cannot distinguish them.

A new authorized query can also invalidate the old capability. Adding query mask 1 requires a fourth independent measurement and sixteen distinguishable retained states. Worlds 7 and 8 agree on all three measurements in the original design but disagree on this new query. The earlier artifact's contract binding fails before its old decoder is treated as appropriate for the amended task.

If the purpose really requires every coordinate, there is no reduction from eight exact probes. If programmable measurements are unavailable, the transformation is not realizable under the existing permission and interface boundary. These are successful refusal or requalification cases, not inconvenient examples to remove.

Noise changes the model as well. With independent flip probability $1/20$ on each of the three measured basis bits, the unchanged decoder has full-vector accuracy

$$(19/20)^3 = 6859/8000 = 0.857375, \tag{10}$$

not 1. Repeating each basis measurement three times and taking a majority would raise this particular independent-noise calculation to approximately 0.978407, but would use nine probes rather than three. Neither figure extends the noiseless certificate. Unknown or correlated noise requires its own model and validation.

4.6 The best game depends on what its moves cost

Assign price 1 to a coordinate probe and price $\alpha > 0$ to a noncoordinate probe. Exhaustive coverage analysis over the fifteen permitted masks gives the nondominated count pairs

$$(c, p) \in \{(8, 0), (6, 1), (4, 2), (0, 3)\}. \tag{11}$$

Here nondominated refers to coordinate and noncoordinate counts separately. For a linear common-price objective, the minimum acquisition price is

$$\min\{8, 6 + \alpha, 4 + 2\alpha, 3\alpha\} = \min\{8, 3\alpha\}. \tag{12}$$

The middle points never uniquely minimize this scalar price. At $\alpha = 1$, the three-probe design costs 3 units. At $\alpha = 3$, coordinate acquisition costs 8 while that design costs 9. Both remain available; a competent selector should change its choice. These are assumed probe prices, not measured economic costs.

For noiseless linear probes and an exact all-world requirement, adaptation cannot improve the minimum in Equation (12): every successful terminal transcript must contain a set of probes whose span contains the query span. Every such terminal set costs at least the cheapest sufficient set, which a nonadaptive protocol can use directly. This argument does not extend to arbitrary nonlinear, noisy, or stochastic-cost experiments.

4.7 What was and was not discovered

The finite constructor finds a purpose-sufficient probe set and decoder within a supplied finite class. It is not handed the winning basis. But the investigator supplies the world family, the purpose, the probe grammar,

and the permission to search for a different protocol. It does not discover a natural-world causal model, invent an instrument, learn the user's values, or call a frontier model.

Five permutations of query and probe order all return adequate three-probe protocols, with different search counts and sometimes different bases. Tests also cover all 127 nonempty query families in a three-bit world against an ordinary rank-based control. These are exhaustive small-model and order-dependence checks, not independent natural-task discovery campaigns.

The result is deliberately interpretable: useful reformulation may require *less world information, a better information channel, and a less restrictive intermediate objective together*. The next research problem is to learn when comparable opportunities exist without receiving such a complete, convenient model.

5 An architecture that discovers formulations rather than decorates prompts

5.1 Start with the obstruction, not a catalogue of tricks

The system begins by constructing an evidence-linked account of what prevents the desired outcome. Each apparent restriction is tagged as a required end, a hard constraint, a physical or environmental fact, an uncertain hypothesis, or an inherited design choice. The classification is revisable; a claim that something is inherited is not evidence that removing it is harmless.

The diagnosis should distinguish at least four explanations for failure: an inadequate solver, insufficient information, an unnecessarily restrictive setup, and a genuinely infeasible requirement. Repeated failure alone does not identify which applies. In the finite study, indistinguishable worlds give an information witness; the factorial comparison identifies a pair of complementary restrictions. In natural work, a witness may instead be a reproducible dependency bottleneck, a measured information loss, or a counterexample to an assumed intermediate requirement.

The practical output is a *restriction hypothesis*, not a statement that the system has found the ultimate formulation. It names the alleged restriction, the purpose it supposedly serves, the evidence for its necessity or dispensability, the alternative, and an observation that could falsify the proposal.

5.2 Search for executable formulation–experiment pairs

A proposal should contain both an alternative construction and a discriminating test. For a claimed invariant, the test searches cases that distinguish the invariant from an accidental pattern. For a changed coordination rule, it holds resources and required outputs constant while testing the dependency mechanism. For a new information protocol, it specifies which possible observation would separate the competing explanations.

This extends ordinary value-of-computation reasoning (Hay et al., 2012): the next action might create useful evidence, build a representation, change an interface, or ask for an indispensable clarification rather than merely run another reasoning pass. Estimated experiment value remains uncertain. The architecture is not granted an oracle telling it which test will be decisive.

5.3 Preserve complementary candidates without promoting unready systems

A representation may need a compatible solver. A new instrument may be useful only after an unnecessarily strong intermediate objective is removed. A certificate-producing artifact may need a corresponding checker. Therefore development should maintain a bounded portfolio of partial alternatives and explicitly test predicted combinations.

The main study's 8, 8, 8, 3 interaction is a specific example. It does not establish that arbitrary combinations should be explored or that temporary regressions are desirable. A rational development policy can discard most candidates, retain a few with a stated mechanism, and stop when the information or construction budget

is exhausted. Development tolerance is not deployment tolerance: a promising incomplete alternative is not an approved operating method.

5.4 Permit new operators without silent new authority

A fixed catalogue of transformations can become the next restrictive formulation. A general implementation should therefore support proposals for new task representations, executable operators, and domain-specific checks. That extension must be explicit and versioned. Model-generated code should run only through admitted tools and a suitably isolated environment; imported research results are data, not authority.

The comparison baseline must receive the same authorized opportunity to introduce relevant methods. Otherwise apparent "transformational intelligence" may only measure a privileged tool grant. The ability to propose changes to an operator registry is distinct from permission to install them, spend funds, act on third-party systems, or alter a deployed service.

Algorithm 1 Discover a better formulation and establish its scope

Require: Purpose contract, incumbent formulation, source evidence, resource limits, retained methods
Reproduce the incumbent and identify missing outcome or cost measurements
Diagnose a possible binding restriction; distinguish required ends from inherited means
Include retaining the incumbent, reusing a method, and stopping among the options
Generate executable formulation–experiment pairs, including justified combinations
for each selected candidate within the exploration budget **do**
    Check authority before construction or information acquisition
    Run development tests; preserve actual artifacts, costs, failures, and unresolved gaps
end for
Construct the finalist's outcome bridge, validity conditions, and recovery procedure
Freeze the finalist and comparable baselines before fresh confirmation
Check the original outcomes and test the proposed structural mechanism
Classify the result: invalid, inconclusive, supported within scope, or a certified bound escape
Promote only the separately approved operating scope; preserve the incumbent otherwise
Reuse conditionally; test discovery-policy changes through fresh complete campaigns

5.5 Make the user interface express the scientific distinction

The user should be able to state a desired outcome without first inventing its best formulation. A useful interface then shows four things: the restriction being questioned, the different setup proposed, the actual artifact or experiment produced, and what the evidence does or does not support. The default should not be a list of impressive-sounding strategies whose implementation never occurs.

For the book example, distinguish a *fixed-summary* contract from an *understanding* contract before replacing the artifact. For mathematics, distinguish a conjecture, a finite computational check, a reviewed argument, and a formally checked proof. For Computer Use, distinguish observed screenshots from executed code and independently checked final state. These distinctions should guide construction without becoming the product's entire experience. The central experience remains discovery of a useful alternative.

6 A better game must still work tomorrow

6.1 Persistent capability changes the future problem

A transformation can produce more than a single answer. It can leave a representation, instrument, program, or coordination mechanism that changes the cost or feasibility of future work. DreamCoder, executable skill libraries such as Voyager, and self-adjusting computation provide technical precedents for retained

abstractions, programs, and maintained results (Ellis et al., 2020; Wang et al., 2023; Acar et al., 2011).

However, the information sufficient for today's output is not necessarily sufficient for tomorrow's obligation. The query study's added mask already shows this. A dynamic system requires preservation under future events, not just under repetitions of a static query family.

6.2 Retained study: a sufficient controller, not merely a small one

The companion retains and reruns the previous edition's deterministic service-monitor experiment. Its source has r required Boolean conditions and n nuisance bits; an event toggles one bit. The service must report initially and after every event whether all r required conditions hold. The constructor receives an anonymized transition/output table and applies standard partition refinement.

For $r = 4, n = 6$, it constructs a 16-state controller from a 1,024-state source. A map ϕ from source states to controller states is checked against

$$\phi(s_0) = q_0, \tag{13}$$

$$o(s) = \bar{o}(\phi(s)), \tag{14}$$

$$\phi(\delta(s, a)) = \bar{\delta}(\phi(s), a) \qquad \text{for every state–event pair.} \tag{15}$$

The separate checker verifies 11,265 local semantic equalities. Induction gives every required output after every finite event sequence in the supplied model. This is a standard finite-state preservation construction, related to classical minimization and bisimulation (Hopcroft, 1971; Dean and Givan, 1997); it is not a new minimization algorithm.

A two-state shortcut that keeps only the current answer fails after four events in the retained main instance. Requiring one additional audit bit doubles the sufficient state count to 32; requiring the entire source state prevents compression. The old study's 54 tests pass again. Appendix B states the exact equivalence and lower-bound conditions.

Table 4: Two exact laboratories, two different restrictions. Neither reports a frontier-model result.

	New query-design study	Retained dynamic-state study
What changes?	Information protocol and intermediate objective together	Retained representation and its transition rule
What remains required?	Every answer in the authorized static query family	Initial output and every required event response
Exact result	Eight required probes become three after the joint change	1,024 source states become 16 sufficient states
Tempting invalid shortcut	Two probes; or an unannounced extra query	Keeping only the present ready/not-ready answer
Strong control	Ordinary linear algebra matches the three-probe design	Relevant-state projection matches the minimal controller
Not established	Total compute saving or general experiment invention	A 64-fold memory, runtime, or economic saving

6.3 Preserve required futures, not every imaginable future

A capability is valid relative to a declared event and question scope. No lossy representation can answer every possible future question about its source. If two distinct source states map to the same retained value, a binary question that distinguishes those states cannot be answered correctly on both using only the retained value. This simple argument explains why an unexpected legitimate query can require recovery or requalification.

Compact operational state is different from retained source data, audit evidence, and recovery material. Reducing the first does not authorize deleting the others. Similarly, a new task family may be a valuable opportunity, but success on it must not be counted as progress while obligations from the original family are quietly abandoned.

A useful capability record therefore contains the changed restriction, executable artifact, supported scope, original-outcome bridge, costs, preconditions, counterexamples, and retirement triggers. The router that chooses and combines such records is another policy requiring evaluation. A library of individually valid artifacts does not automatically compose into a valid operating system.

7 Pay for discovering and sustaining the alternative

7.1 A probe advantage is not a lifecycle advantage

A measurement protocol can reduce world-dependent interactions while increasing offline computation. A compact controller can require substantial construction and retained evidence. A clever interface can move work to a reviewer. Report those quantities separately before combining them under an explicitly justified cost model.

For transformation T, let K_T include discovery, failed candidates, construction, confirmation, migration, and initial admission. Let L_T be its valid-use lifetime, and let s_t be its per-use advantage after execution, verification, maintenance, and displaced costs. A simple common-unit value model is

$$V_H(T) = -K_T + \mathbb{E}\left[\sum_{t=1}^{H} \mathbf{1}\{L_T \geq t\} s_t\right]. \tag{16}$$

This expression assumes that the incumbent resumes after invalidation without further difference; recovery losses, retirement costs, or changed post-invalidation performance must be added if present. Context-dependent gains and survival cannot generally be separated into independent averages.

If the first use is valid, $s_t = s \geq 0$, and $\mathbb{P}(L_T \geq t) = p^{t-1}$, then

$$V_H = -K + s\frac{1 - p^H}{1 - p} \quad (p < 1), \qquad V_H = -K + sH \quad (p = 1). \tag{17}$$

This follows by summing a geometric series. For $p < 1$, $K \geq s/(1 - p)$ prevents strictly positive value at every finite horizon. The companion reruns the earlier exact arithmetic. These are illustrative assumptions, not a fitted reliability or commercial model.

The lesson is not "always construct a reusable system." It is to compare retaining the incumbent, using an adequate existing capability, acquiring more evidence, and building a new one. A transformation can be intellectually elegant and operationally uneconomic. Correctly declining it is evidence of judgment.

7.2 Freeing a resource is an opportunity, not automatically a second outcome

Resources no longer needed by the incumbent might support additional authorized goals. To claim capability expansion, actually execute the additional work under the same total budget and confirm that previous obligations remain satisfied. Savings forecast from a task counter are not a demonstration that a new scientific result or useful service has become feasible.

Shared infrastructures complicate attribution. Two transformations may depend on the same investment, invalidate one another, or claim the same saved operation. Evaluate the combined trajectory and count common construction costs once. Do not sum independently advertised gains when the methods cannot operate together.

8 An empirical program that cannot win by moving the goalposts

8.1 The target is discovery, not supplied ingenuity

The finite examples establish opportunities and their limits under explicit models. They do not establish that an LLM can recognize comparable restrictions in unfamiliar natural tasks. A natural-task study should begin with a useful purpose, source material, and permitted actions, without giving the agent the winning abstraction or telling it which restriction is removable.

Record what the investigator supplied: contract, world model, candidate operators, semantic annotations, solution hints, test generator, and previous artifacts. A method that executes a supplied transformation has demonstrated realization; one that independently selects it from a finite supplied class has demonstrated class-relative discovery. Generalizing that discovery process to new mechanisms requires further evidence.

A strong study should include both independently selected natural workloads and synthetic tasks whose limits are known. The natural workload tests relevance. The synthetic model supplies exact negative cases and upper bounds. Neither should be represented as the other.

8.2 Comparators and a mechanism test

Use the same underlying model access, external permissions, relevant initial knowledge, and declared resource envelope where practical. Compare the incumbent, a strong solver within its formulation, a strong unrestricted agent explicitly allowed to reframe, a relevant domain-engineered method, and applicable automated agent or workflow search. ADAS, AFlow, and GEPA supply concrete comparison families rather than a straw-man one-shot prompt (Hu et al., 2024; Zhang et al., 2024; Agrawal et al., 2025).

Cross formulation changes with compatible solver changes. Test single changes and predicted combinations, as in Table 2. Give the discovered formulation to the old solver as a diagnostic control. Give the proposed discovery machinery the old formulation as another control. These tests distinguish the value of the setup, the quality of its implementation, and the ability to discover it.

An ablation that destroys the only compatible solver is not automatically evidence against a new representation. Conversely, a representation label attached to an otherwise unchanged program is not structural attribution. The mechanism hypothesis should predict which interventions matter and which should not.

8.3 Fresh confirmation and independent outcome grounding

Select a finalist and freeze model configuration, tool versions, source bindings, and confirmation rules before using fresh terminal cases. Within-episode adaptation may remain part of the evaluated policy; cross-episode learning must be disabled or evaluated at a larger unit. Different questions from one book or tasks from one repository are not automatically independent replicates.

Repeated feedback can make a holdout part of development rather than confirmation (Dwork et al., 2014). Sequential claims need statistical methods with assumptions matching the actual procedure (Howard et al., 2021). Exact finite checking needs no sampling interval, but remains conditional on the source model and trusted implementation. Statistical confidence in a proxy is not confidence that the proxy measures the desired outcome.

A numerical comparison should prespecify its outcome, minimum meaningful difference, sampling unit, and complete cost boundary. Use a disjoint pilot to estimate variability and plan confirmation effort; do not choose sample size retrospectively because the desired result has appeared. Count all confirmatory attempts, including unsuccessful ones, and report uncertainty rather than equating a positive point estimate with a demonstrated advantage.

8.4 The evaluator may be revised, but not selected by the winner

An evaluator can encode the incumbent's unnecessary restrictions. Requiring the old internal trace or a preferred artifact format can reject a valid alternative. The proposed correction must be separately justified using outcome-grounded evidence, including cases where the old and new instruments disagree.

After approval, compare both candidates under the common validated instrument or an independently supported bridge. Retain the previous observations as a different measurement epoch. A candidate should not receive credit simply because its chosen judge is easier to satisfy. Where the intended outcome is still ambiguous, clarification is substantive work, not a measurement trick.

8.5 Six falsifiable hypotheses

H1: useful formulation discovery. At matched lifecycle resources, the proposed system discovers alternatives that improve authorized outcomes on fresh tasks against strong unrestricted comparisons. The claim is not supported if the advantage depends on extra tools, uncharged engineering, or a hidden change of objective.

H2: complementary structural change. Some gains require predicted combinations of representation, information, or intermediate-objective changes. The mechanism claim weakens when a comparably tuned single change explains the entire gain. The finite factorial result is evidence for one constructed instance, not a prevalence estimate.

H3: selective recognition. The system distinguishes removable assumptions from genuine obligations and from restrictions whose removal is too costly. It must find useful transformations and reject inappropriate ones. Universal refusal and universal reframing both fail this test.

H4: better evidence acquisition. The system identifies and executes experiments that improve decisions after acquisition and construction costs are counted. More hypotheses, more citations, or persuasive post hoc narratives do not establish this result.

H5: valid persistence and expansion. Retained capabilities remain appropriate under the declared future scope, detect relevant changes, and support measured reuse or additional outcomes without sacrificing previous obligations. The claim fails when saved effort merely delays an error or shifts work outside the accounting boundary.

H6: improvement of the discovery process. A changed discovery policy produces better outcomes over independent complete campaigns on unfamiliar mechanism families, from matched starting agents and libraries. The claim fails if the apparent improvement comes only from inheriting a superior artifact or repeated exposure to the same evaluation cases.

8.6 A concrete natural-task program

A first mission could investigate a public computational service whose repeated workload is expensive but whose outputs can be checked independently. The contract should specify the real deliverable, performance objective, accuracy, update semantics, resource limits, and requirements that must not change. The agent is allowed to inspect public sources, run permitted experiments, construct alternative representations or algorithms, and retain a tested capability.

A scheduling service is one possible domain: an agent could discover that a work decomposition or representation unnecessarily constrains performance, while an independent checker preserves required tasks, durations, resources, and genuine dependencies. OR-Tools is an established implementation baseline, not something to omit to create an artificial victory (Google, 2026). The service contract must distinguish optional internal decomposition from dependencies that are physical or contractual.

For scientific investigation, a useful mission could require a mechanism hypothesis and the experiment that discriminates it from the strongest alternative. For mathematics, it could seek a general construction, invariant, or reduction, with exact assumptions and independent proof assessment. Novelty, validity, and practical importance are separate questions. Neither the present finite computations nor their tests settle any open natural-domain problem.

For computer-mediated work, environments with independently inspectable final states, such as the OSWorld research setting, provide relevant evaluation foundations (Xie et al., 2024). Computer Use should expand access to permitted tools, not freeze a task into unnecessary clicks. Where an API or compiled program is permitted and better, the system should be allowed to discover and demonstrate that change.

9 SUCCESSOR Ω: learn to discover better games

9.1 Retain a mechanism with its conditions

The long-term object is not an indefinitely growing transcript. It is a conditional collection of capabilities and evidence about when they help. Retain the binding restriction, alternative representation or process, executable artifact, outcome bridge, construction cost, supported contexts, known failures, and requalification conditions.

A query decoder is not valid for arbitrary future questions. A service controller is not valid under arbitrary new outputs. An experiment is not informative merely because it distinguished hypotheses in a different system. The library should make these distinctions operational, including the option not to reuse.

9.2 The discovery policy is another candidate

Let η select investigations, candidate formulations, experiments, combinations, and resource allocations. A complete discovery campaign starts from initialization Z and a fixed initial library, uses budget B, and returns a candidate system $\mathcal{K}_{\eta,Z,B}$. A terminal evaluation then measures its longitudinal utility and operating costs on fresh conditions. A campaign-level objective is

$$\mathcal{V}(\eta) = \mathbb{E}_Z \left[J_{\text{terminal}}(\mathcal{K}_{\eta,Z,B}) - C_{\text{discovery}} \right], \tag{18}$$

where cost and utility must be commensurable or kept as separate constrained objectives. Terminal costs must not be subtracted twice. Failed candidates, construction, reviewer effort, and investigation belong in the campaign ledger.

Compare old and new discovery policies across fresh *complete* campaigns. Cross old and new starting libraries when needed to distinguish better discovery from better inheritance. Hundreds of observations inside one campaign are not hundreds of independent demonstrations that the discovery policy improved.

There are direct precedents: the Darwin Gödel Machine evaluates self-modified agents, and Hyperagents makes the meta-level modification procedure editable (Zhang et al., 2025, 2026). These systems are comparison targets, not evidence that the present manuscript has already delivered general self-improvement. The question here is whether learned changes improve the discovery of useful, purpose-preserving formulations.

9.3 Compounding is possible; endless acceleration is not established

A better representation can make future alternatives easier to express. A validated checker can lower the cost of evaluating them. An effective experiment policy can reduce wasted construction. Such changes may reinforce one another. They can also produce brittle dependencies, rising maintenance costs, or a library too difficult to route correctly.

No monotonic or exponential growth law follows from the architecture. A fixed bounded outcome cannot improve by a fixed positive amount indefinitely. Continuous discovery means an enduring ability to investigate, test, retain, revise, and sometimes stop. New objectives or environments create new comparisons, not permission to concatenate incompatible scores into a single acceleration curve.

10 An implementation path that preserves the ambition

The proposed target is `MontrealAI/AGI-Alpha-Agent-v0`. The interpretation of the mission is concrete: *Find your Alpha* means discovering a valuable restriction and a feasible way to challenge it; *Build your SUCCESSOR* Ω means constructing a useful alternative and retaining it with evidence and conditions of reuse. This manuscript does not modify or release that repository.

A general implementation should use one mission history and a versioned task-adapter boundary. Each adapter should identify its input contract, executable proposal language, permitted tools, outcome checker, cost ledger, validity assumptions, and recovery path. The present finite constructor and checker are local research components, not already qualified product integrations.

Table 5: Proposed implementation responsibilities. These are design requirements, not claims of delivered repository functionality.

Responsibility	Practical requirement
Discover the restriction	Source-linked assumption map with genuine requirements separated from proposed means
Construct a different game	Admitted model/tool adapter that produces an executable alternative, not only a description
Invent the decisive test	Explicit predictions, allowed observations, and a comparison that can reject the mechanism
Check the outcome	A domain-specific bridge evaluated outside the candidate's preferred scoring path
Retain a capability	Versioned artifact with scope, costs, known failures, and recovery conditions
Improve discovery	Matched independent campaigns with fresh terminal evaluation and complete failed-attempt accounting

Alpha would organize discovery and construction. Jobs would commission bounded work and review under separately admitted terms. Fleet would support qualified execution and evidence transport. Useful discovery does not become true because a receipt is signed or a payment route exists. Conversely, an exact finite result need not await a financial deployment to be scientifically inspectable.

Preserve existing meanings when adding task types. A zero-money research request does not authorize paid model inference or settlement. USDC and AGIALPHA routes must remain distinct. Model proposals, imported results, saved histories, and hashes do not carry new authority. Publication, live operation, and financial activation are separate events. These boundaries protect a meaningful implementation; they should not displace the central work of discovering and building the alternative.

The first meaningful implementation milestone is one actual model-driven public-input mission: a diagnosed restriction, executed candidate, strong same-permission comparisons, an independently inspectable

outcome, a complete cost record, and a future case where reuse should be refused. A result where the strong baseline wins is still a valid experiment. A static interface that narrates an unexecuted discovery is not the milestone.

11 Relation to prior work and remaining limitations

11.1 The conceptual and technical landscape

Transformational creativity and abstraction. Boden's distinction between exploratory and transformational creativity supplies the conceptual precedent; Ritchie emphasizes the need to operationalize claims about transformed spaces (Boden, 1998; Ritchie, 2006). Automated abstraction work such as ALPINE shows that changing a representation has long been a technical research program (Knoblock, 1994). The contribution here is not the slogan "change the search space" but a purpose-linked decomposition of the claim, with concrete constructions and controls.

Information and experimentation. Blackwell's comparison of experiments and modern experimental-design methods connect information structures to decisions (Blackwell, 1953; Rainforth et al., 2024). BED-LLM explicitly uses experimental design for adaptive information gathering by language models (Choudhury et al., 2025). Action-BED jointly optimizes experiment-design and downstream-action policies against task loss (Rossa et al., 2026). These are particularly relevant precedents: neither task-directed measurement nor coupling measurement to the eventual decision is claimed as new here. The query laboratory instead provides a transparent finite factorial, an exact restricted frontier, and independently checkable outputs inside the broader formulation-discovery framework.

Programs, representations, and skills. DreamCoder, Voyager, and self-adjusting computation provide foundations for learned abstractions, retained executable skills, and maintaining results as inputs change (Ellis et al., 2020; Wang et al., 2023; Acar et al., 2011). The retained controller study uses established finite-state ideas (Hopcroft, 1971; Dean and Givan, 1997). Separating construction from a checkable certificate also has a long history, including proof-carrying code (Necula, 1997). Our small certificates do not inherit the security properties of an entire verified system.

Algorithmic discovery and agent search. AlphaEvolve uses model-generated program changes, automated evaluation, and evolutionary search in scientific and computational applications (Novikov et al., 2025). ADAS, AFlow, and GEPA search agent structures, workflows, or prompts (Hu et al., 2024; Zhang et al., 2024; Agrawal et al., 2025); DGM and Hyperagents expose increasingly editable improvement procedures (Zhang et al., 2025, 2026). Any such method may discover a transformational change depending on what it actually constructs. The category is not exclusive to a particular product or algorithm.

11.2 What this manuscript does not establish

Both laboratories are small, exact, finite, and investigator-designed. The query constructor searches a supplied operation class with complete model semantics. The controller constructor receives a complete deterministic transition model. Neither learns a natural model, invents an unprovided instrument, makes a frontier-model call, or solves an open real-world research problem. Strong ordinary methods match both constructions.

The query result depends on a static world, noiseless measurements, an explicit output family, and the stipulated probe interface. The controller result depends on complete and correct event/output semantics. The new checker and retained checker are separately implemented, but not organizationally independent. Passing tests validates specified behavior, not the adequacy of every scientific or operational assumption.

No measured total runtime, energy, memory, revenue, saved human time, or production benefit is claimed. The probe frontier and state counts are exact quantities with narrower meanings. Prices and survival probabilities are explicit assumptions. Unmodeled side effects, missing observations, environmental drift, and ambiguous human purposes can all invalidate an apparently elegant transformation.

These limits identify the next experiments. The strongest next result is not a more elaborate synthetic trick. It is a useful natural mission whose restriction was not supplied as the answer, where an executable alternative survives meaningful comparisons and a fresh test of its claimed mechanism.

12 Conclusion: discover the formulation in which progress becomes possible

A system can be excellent at a policy and still be confined by the problem it believes it must solve. The original formulation may demand unnecessary information, preserve avoidable dependencies, impose an intermediate reconstruction, or convert a reusable capability into endless repeated work. More effort within that setup need not remove its restriction.

The exact query study makes a specific interaction visible. Neither changing sensing alone nor relaxing an inherited reconstruction target alone improves the exact eight-probe requirement. Together they make all seven required answers attainable with three probes. A finite constructor finds an adequate protocol, a separate checker validates its outputs, and an unrestricted ordinary method matches the result. The dynamic study adds the necessary counterweight: a representation that is correct now can still discard what future obligations require.

The broader objective is not to minimize task counts, maximize reframing, or create an agent population that never stops working. It is to build an AI that recognizes when the apparent problem is unnecessarily restrictive, discovers a better setup, constructs the means to use it, and demonstrates that the change serves the real purpose.

Do not only build intelligence that finds a better answer to the original problem. Build intelligence that discovers a better problem-solving setup—and can show why the different game is worth playing.

Research status and revision record

This is a revised and retitled edition of *Beyond Task Completion*. It broadens the central thesis from eliminating recurring work to discovering purpose-preserving formulations that expand attainable outcomes. The original *Transformational Intelligence: Learning to Change the Game* manuscript and its public link remain unchanged. The earlier dynamic-state source and tests are retained separately and rerun.

The companion contains analytic arguments, a new exact query-design study, the retained exact controller study, explicit cost illustrations, and a proposed empirical program. The executed suites pass 45 new tests and 54 retained tests, for 99 tests in total. The programs use Python's standard library, make no network or model calls, require no credentials, and perform no repository writes, payments, or deployments. No external peer review or general transformational-intelligence achievement is claimed. Literature was checked on October 19, 2026.

A Proof details and interpretation of the query frontier

A.1 Sufficiency of a probe set

Let P be a matrix whose rows are selected probe masks. The observation is $P\theta$, and the required vector is $Q\theta$. The exact condition for a deterministic answerer is

$$\ker P \subseteq \ker Q, \tag{19}$$

equivalently $\text{rowspan}(Q) \subseteq \text{rowspan}(P)$. If Equation (19) holds, observations equal on worlds θ and θ' imply equal required outputs, so the decoder is well defined. If it fails, choose $v \in \ker P \setminus \ker Q$: worlds 0 and v have the same observations and different required vectors. This is a constructive counterexample to any decoder based only on those observations.

The table constructor checks exactly this functional condition by enumerating worlds. The strong algebraic comparator checks the row-space inclusion using ordinary elimination. Agreement between the two is a useful implementation cross-check; neither is a new linear-algebra algorithm.

A.2 Why adaptive coordinate selection does not help this instance

After an adaptive coordinate transcript, let S be the distinct observed coordinates and x_S their values. With independent uniform initial coordinates and noiseless coordinate reads, the transcript fixes only those coordinate values. The choices of which coordinates to read are functions of earlier observations and independent internal randomness. Conditioning on a fixed random seed and transcript therefore leaves the unread coordinates uniform.

Consequently, $Q\theta$ is uniform on an affine translate of the image of $Q_{\overline{S}}$, with support size $2^{\text{rank}(Q_{\overline{S}})}$. The best full-vector guess has conditional success $2^{-\text{rank}(Q_{\overline{S}})}$. Every possible terminal S with $|S| \le b$ has success at most the maximum in Equation (9); averaging cannot exceed it. A nonadaptive maximizing set attains the maximum. Repeated reads and early stopping cannot improve the at-most-b bound.

For the basis rows in Equation (6), the eight coordinate columns consist of four distinct nonzero three-bit patterns, each occurring twice. Any three distinct patterns in this collection are linearly independent. With at most three coordinates read, at least five remain, containing at least three different patterns; the residual rank is three. Reading four coordinates can leave exactly two patterns, giving rank two. Reading six can leave exactly one pattern, giving rank one. Reading all eight gives rank zero. This yields Table 3, with the enumeration serving as an independent numerical check of the cases.

A.3 Why adaptive linear probes cannot beat the exact price minimum

Fix a transcript of an exact successful protocol. The probe masks on that path form a matrix P and constrain the possible worlds to a nonempty affine coset of $\ker P$. If $\ker P$ is not contained in $\ker Q$, that coset contains two worlds with different required vectors and identical transcript responses. They follow the same adaptive path because every prior response is identical. The protocol could not be correct on both.

Every successful terminal path must therefore contain a sufficient probe set. With positive additive probe prices, every path costs at least the minimum price of such a set. A fixed cheapest sufficient set attains this bound. The exhaustive catalogue analysis enumerates the nondominated sufficient-set count pairs rather than assuming that the three-probe design is cheapest at every price. Discovery and verification costs are outside this *acquisition-only* optimization and are charged separately in a lifecycle comparison.

A.4 The certificate and its trusted assumptions

The expected contract contains the dimension, query masks, allowed probe masks, world-model identifier, and required-output identifier. A capability supplies a contract digest, probe list, and a decoder for reachable

response values. The checker rejects unknown fields, malformed masks, duplicate probes, noncanonical decoder keys, stale contract bindings, missing or extra decoder rows, and incorrect outputs.

For every one of 2^d worlds it computes the observation and verifies every one of $|Q|$ required bits. Thus a successful check proves the decoder's stated finite semantic property under the input model. It does not authenticate the producer's search history, establish that the expected contract was authorized by a human, or prove that a physical sensor implements the ideal probe. A digest is a byte binding, not semantic truth or authority.

B The retained dynamic-state result

B.1 Future-obligation equivalence

For a finite deterministic service machine $M = (S, \Sigma, \delta, o, s_0)$, define

$$s \sim_{\mathcal{I}} t \quad \Longleftrightarrow \quad o(\delta^*(s, w)) = o(\delta^*(t, w)) \quad \text{for all } w \in \Sigma^*. \tag{20}$$

The empty word preserves the current required output; all other words preserve future outputs. Equivalent states transition to equivalent states under each event, so the quotient is a well-defined deterministic output machine. The map conditions in Equation (15) prove output preservation by induction on word length.

When all source states are reachable, any smaller deterministic retained-state controller with the same required event/output behavior and no uncounted external memory would merge two inequivalent reached states. A common future word distinguishes their required outputs, contradicting correctness. This is the standard minimum-state argument under the stated memory and observation boundary.

For the monitor with r required bits, retaining those bits suffices. Distinct required-bit vectors u, v are future-distinguishable: toggle exactly the positions that are zero in u. The resulting vector from u is all ones; the resulting vector from v is not. There are therefore 2^r necessary controller states, regardless of the number of irrelevant nuisance bits.

B.2 Retained numerical results

The main instance has 1,024 source states, ten event types, and 16 quotient states. The source and controller transition tables have 10,240 and 160 cells. The live state indices need ten and four bits, respectively. This is not a 64-fold live-memory reduction; the state-index reduction is six bits. Source models, maps, verification records, and recovery material remain stored in the evidence package.

The original 10,000-event replay compares 10,001 required outputs. The preserved weak output-only control fails on the four-event word $(0, 1, 4, 7)$ under the retained label permutation: the source outputs $(0, 0, 0, 0, 1)$ and the shortcut outputs $(0, 0, 0, 0, 0)$. A full-state output contract yields 1,024 necessary states; a readiness-plus-one-audit-bit contract yields 32.

The earlier sweep covers eight parameter settings and five identifier relabelings each. It checks 183,580 certificate equalities and compares 40,040 replay outputs. These are regression cases for one specified model family, not forty independent demonstrations of learned discovery. The separate checker shares the source-model assumptions and the artifact's authorship.

C Reproduction and evidence package

C.1 Run both laboratories

From the extracted project directory, using Python's standard library:

```
python experiments/run_game_study.py
python -m unittest discover -s tests -v
python experiments/check_game_artifact.py \
  results/query_contract.json results/query_capability.json
cd legacy
python experiments/run_study.py
python -m unittest discover -s tests -v
python experiments/check_artifact.py results/capability.json
```

The new suite contains 45 test methods, including an exhaustive small-family subtest; the retained suite contains 54. Their outputs are saved separately. Changing probe order is not a fresh scientific campaign. JSON files are inert research data and are not general code-execution adapters.

C.2 What each artifact supports

File or directory	Role
`experiments/game_discovery.py`	Finite contract, table constructor, algebraic control, frontier and cost calculations
`experiments/verify_game.py`	Separate exhaustive checker against an expected contract
`experiments/run_game_study.py`	Deterministic new-study driver and saved observations
`experiments/check_game_artifact.py`	Bounded local JSON verification command
`results/query_contract.json`	Expected finite purpose and permitted interface
`results/query_capability.json`	Constructed three-probe protocol and decoder
`results/game_study.json`	Factorial results, frontier, counterexamples, price choices, and scope controls
`legacy/`	Retained dynamic-model sources, tests, results, and checker
`CLAIMS.md`	Supported claims, evidence, and explicit exclusions
`REVISION_NOTES.md`	Conceptual and empirical differences from the previous edition
`MANIFEST.sha256`	File-integrity manifest; not a signature or semantic proof

The standalone LaTeX source embeds its tables and bibliography and needs no external figure or font file. Compile with `latexmk -pdf Beyond_the_Original_Problem.tex`, or run `pdflatex` twice. The numerical results are generated by the Python programs, not inferred from the typeset tables. The original *Transformational Intelligence* and first-edition *Beyond Task Completion* files are not overwritten.

C.3 Claim–evidence register

Claim	Evidence	Scope and exclusion
Task counts can change without outcome change	Label-refinement proposition	Internal labels, not externally prescribed obligations
Purpose can require less information than full-world reconstruction	Theorem 4.1; decoder construction	Supplied finite model and fixed query family
Two structural changes are jointly useful when each alone is not	Exact $8, 8, 8, 3$ factorial and lower bounds	Probe counts; not a claim about prevalence in natural tasks
At three probes, the coordinate-only optimum is $1/8$ and the purpose-directed optimum is 1	Complete coordinate enumeration, tight bounds, executable design	Uniform noiseless world; entire-vector success; acquisition budget only
A finite procedure constructs an adequate protocol	502 tested candidate sets and portable decoder	Finite-class search, not autonomous natural-model or instrument invention
All seven outputs are correct in every modeled world	Separate checker; 1,792 output equalities	No claim of organizational independence or physical sensor verification
An unrestricted ordinary method matches the protocol	Algebraic minimum and row-space certificate condition	No superiority claim over standard linear algebra
Additional obligations can invalidate sufficiency	Added-query witness; four-probe result	New contract requires a new qualification
Costs can reverse the preferred protocol	Complete finite price frontier	Assumed probe prices; excludes total construction costs
A compact present answer may fail in the future	Retained four-event counterexample and quotient certificate	Specified deterministic event model
General AI can learn to discover better formulations	H1–H6 and matched-campaign design	Hypothesis; no frontier-model or natural-task study in this artifact

References

Umut A. Acar, Matthias Blume, and Jacob Donham. *A Consistent Semantics of Self-Adjusting Computation*. arXiv:1106.0478, 2011. https://arxiv.org/abs/1106.0478.

Lakshya A. Agrawal et al. *GEPA: Reflective Prompt Evolution Can Outperform Reinforcement Learning*. arXiv:2507.19457, 2025. https://arxiv.org/abs/2507.19457.

David Blackwell. Equivalent comparisons of experiments. *The Annals of Mathematical Statistics*, 24(2):265–272, 1953. https://doi.org/10.1214/aoms/1177729032.

Margaret A. Boden. Creativity and artificial intelligence. *Artificial Intelligence*, 103(1–2):347–356, 1998. https://doi.org/10.1016/S0004-3702(98)00055-1.

Vincent Boucher. *Transformational Intelligence: Learning to Change the Game*. MONTREAL.AI research manuscript, October 2026. https://github.com/MontrealAI/AGI-Alpha-Agent-v0/blob/main/docs/Transformational_Intelligence.pdf.

Deepro Choudhury et al. *BED-LLM: Intelligent Information Gathering with LLMs and Bayesian Experimental Design*. arXiv:2508.21184, 2025. https://arxiv.org/abs/2508.21184.

Thomas Dean and Robert Givan. Model minimization in Markov decision processes. In *Proceedings of AAAI*, pages 106–111, 1997. https://engineering.purdue.edu/~givan/papers/aaai97-2.pdf.

Cynthia Dwork, Vitaly Feldman, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Aaron Roth. *Preserving Statistical Validity in Adaptive Data Analysis*. arXiv:1411.2664, 2014. https://arxiv.org/abs/1411.2664.

Kevin Ellis et al. *DreamCoder: Growing generalizable, interpretable knowledge with wake-sleep Bayesian program learning*. arXiv:2006.08381, 2020. https://arxiv.org/abs/2006.08381.

Google. *OR-Tools: Constraint Optimization*. Official documentation, accessed October 19, 2026. https://developers.google.com/optimization/cp/.

Nicholas Hay, Stuart Russell, David Tolpin, and Solomon Eyal Shimony. *Selecting Computations: Theory and Applications*. arXiv:1207.5879, 2012. https://arxiv.org/abs/1207.5879.

John E. Hopcroft. *An $n \log n$ Algorithm for Minimizing States in a Finite Automaton*. Stanford Computer Science Technical Report CS-TR-71-190, 1971. https://i.stanford.edu/TR/CS-TR-71-190.html.

Steven R. Howard, Aaditya Ramdas, Jon McAuliffe, and Jasjeet Sekhon. Time-uniform, nonparametric, nonasymptotic confidence sequences. *The Annals of Statistics*, 49(2):1055–1080, 2021. Open version: https://arxiv.org/abs/1810.08240.

Shengran Hu, Cong Lu, and Jeff Clune. *Automated Design of Agentic Systems*. arXiv:2408.08435, 2024. https://arxiv.org/abs/2408.08435.

Craig A. Knoblock. Automatically generating abstractions for planning. *Artificial Intelligence*, 68(2):243–302, 1994. Open author manuscript: https://usc-isi-i2.github.io/papers/knoblock94-aij.pdf.

George C. Necula. Proof-carrying code. In *Proceedings of POPL*, pages 106–119, 1997. https://doi.org/10.1145/263699.263712.

Alexander Novikov et al. *AlphaEvolve: A coding agent for scientific and algorithmic discovery*. arXiv:2506.13131, 2025. https://arxiv.org/abs/2506.13131.

Tom Rainforth, Adam Foster, Desi R. Ivanova, and Freddie Bickford Smith. Modern Bayesian experimental design. *Statistical Science*, 39(1):100–114, 2024. Open preprint: arXiv:2302.14545. https://arxiv.org/abs/2302.14545.

Graeme Ritchie. The transformational creativity hypothesis. *New Generation Computing*, 24:241–266, 2006. https://doi.org/10.1007/BF03037334.

Tom Rossa, Angus Phillips, and Tom Rainforth. *Action-BED: Task-Driven Bayesian Experimental Design with Singly Intractable Objectives*. arXiv:2606.23662v1, 2026. https://arxiv.org/html/2606.23662v1.

Guanzhi Wang et al. *Voyager: An Open-Ended Embodied Agent with Large Language Models*. arXiv:2305.16291, 2023. https://arxiv.org/abs/2305.16291.

Tianbao Xie et al. *OSWorld: Benchmarking Multimodal Agents for Open-Ended Tasks in Real Computer Environments*. arXiv:2404.07972, 2024. https://arxiv.org/abs/2404.07972.

Jiayi Zhang et al. *AFlow: Automating Agentic Workflow Generation*. arXiv:2410.10762, 2024. https://arxiv.org/abs/2410.10762.

Jenny Zhang, Shengran Hu, Cong Lu, Robert Lange, and Jeff Clune. *Darwin Gödel Machine: Open-Ended Evolution of Self-Improving Agents*. arXiv:2505.22954, 2025; revised March 2026. https://arxiv.org/abs/2505.22954.

Jenny Zhang, Bingchen Zhao, Wannan Yang, Jakob Foerster, Jeff Clune, Minqi Jiang, Sam Devlin, and Tatiana Shavrina. *Hyperagents*. arXiv:2603.19461, 2026. https://arxiv.org/abs/2603.19461.