Mathematics from Quantum Randomness

How a QRNG Generated an Object That Led to a Theorem We Did Not Ask For

Douglas C. Youvan

Copyright and Source Note

This book incorporates, reorganizes, and substantially revises material from earlier works by the author, including Quantum Discovery (2024), Structured Quantum Randomness (2025), Pure Information from Quantum Randomness (2025), Mathematical It from Bit (2026), Can Artificial Intelligence Discover Mathematics Before Humans Name It? (2026), Before Algebra Is Named (2026), and The Virgin Oracle (2026). The QRNG experiment chapters are based on the executable repository `qrng-mathematical-abiogenesis`, release v0.2.1.

Historical speculative passages are presented as part of the development of the research program. The exact theorem claims are governed by the executable certificates and theorem report, not by philosophical interpretation.

Preface

This book follows an unusual research path. It does not begin with a theorem and then reconstruct the proof. It begins with a recurring question: can a source of physical randomness participate in the origination of a mathematical question, not merely in the acceleration of a search that humans have already specified? The distinction is easy to blur. Randomized mathematics is old. Monte Carlo methods, randomized algorithms, genetic searches, and pseudorandom exploration can all uncover useful objects. Quantum random number generators add a different physical source of entropy, but replacing one random seed with another does not by itself create a new kind of mathematics.

The experiment described here became interesting only when the role of the random source moved downward. Instead of asking a quantum random number generator to choose among equations that had already been written, we let archived QRNG-origin bytes determine finite operation tables. Those tables were treated as anonymous mathematical worlds. Their internal transformations were measured before they were named or interpreted. One of those worlds produced a striking depth anomaly. That anomaly suggested a question about pairs of transformations on four points. At that moment the randomness became dispensable. Exhaustive mathematics took over and established an exact finite theorem.

The central sentence of the book is therefore deliberately modest: the QRNG did not prove the theorem. It generated an object that caused us to ask a theorem question we had not specified in advance. This is a claim about discovery provenance, not about the logical status of the theorem. The theorem is independently checkable. The historical QRNG provenance is separately qualified. The novelty claim is separately qualified again. Keeping those three layers apart - origin, truth, and priority - is essential to the scientific value of the project.

The chapters draw on and substantially reorganize several earlier works by the author, especially Quantum Discovery (2024), Pure Information from Quantum Randomness (2025), Mathematical It from Bit (2026), Can Artificial Intelligence Discover Mathematics Before Humans Name It? (2026), and The Virgin Oracle (2026). Earlier passages are retained where they document the evolution of the idea, but this book is not a simple anthology. The older proposals are repeatedly evaluated from the standpoint of the later experiment. Some assumptions are narrowed, some speculative interpretations are set aside, and negative results are treated as part of the method rather than hidden.

The result is both a research narrative and a methodological proposal. The mathematical theorem is small enough to verify completely, which is a virtue. It gives us a controlled laboratory for asking a larger question about machine-assisted discovery: what would count as mathematics whose decisive question was not chosen in advance by the human investigator?

Contents

Part I - The Question Before the Experiment

Chapter 1: The Question Before the Experiment

Adapted from Quantum Discovery: Exploring the Search for Mathematical Truths through Quantum Superposition (2024).

In 2024 the project began with a direct but still conventional idea: use quantum computation to search a large space of semi-random mathematical formulae. The following material preserves that proposal as the starting point of the story.

Introduction

Quantum computation, a revolutionary field at the intersection of computer science and quantum physics, harnesses the principles of quantum mechanics to perform computations that would be infeasible with classical computers. Unlike classical bits, which exist in a state of either 0 or 1, quantum bits (qubits) can exist in a superposition of states, allowing quantum computers to process a vast amount of information simultaneously. This capability is further enhanced by quantum entanglement and quantum interference, which enable complex problem-solving and optimization processes far beyond the reach of classical computation. Quantum computation's foundational principles rest on the unique behaviors of qubits.

Superposition allows a qubit to be in a combination of states simultaneously, exponentially increasing the computational power with each additional qubit. Entanglement, a phenomenon where the state of one qubit is dependent on the state of another, regardless of distance, facilitates coordinated and faster problem-solving. Quantum interference allows the correct solutions to be amplified while incorrect ones are canceled out, making quantum algorithms incredibly efficient for specific tasks.

Motivation for Exploring Semi-Random Formulae in Superposition

The exploration of semi-random formulae in superposition presents a novel and

intriguing application of quantum computation. Traditionally, quantum algorithms such as Shor's algorithm for integer factorization and Grover's algorithm for unstructured search have demonstrated quantum computation's potential to outperform classical algorithms in specific areas. However, these applications are rooted in problems with direct analogies to quantum mechanics or classical computational tasks. Our motivation stems from the hypothesis that quantum computation can be extended to search for mathematical truths in a more abstract and exploratory manner. By generating a vast number of semi-random mathematical formulae and placing them in superposition, we aim to leverage quantum computation's inherent capabilities to potentially identify valid and true equations. This approach could not only uncover new mathematical truths but also provide insights into the fundamental nature of mathematical discovery and the relationship between mathematics and quantum mechanics.

Potential Implications for Mathematics and Physics

The potential implications of this research are profound for both mathematics and physics. In mathematics, the discovery of new valid equations through quantum

computation could revolutionize our understanding of various mathematical fields, leading to breakthroughs in number theory, algebra, geometry, and beyond. It could provide new tools for solving longstanding problems, generate novel mathematical structures, and offer fresh perspectives on existing theories. In physics, the application of quantum computation to search for new equations could lead to the discovery of previously unknown physical laws or principles. By exploring the mathematical underpinnings of physical phenomena through this quantum approach, we could gain deeper insights into the fabric of reality, from the fundamental interactions of particles to the large-scale structure of the universe.

Moreover, this interdisciplinary exploration could bridge the gap between abstract mathematical theory and practical physical application, fostering a more integrated understanding of the natural world. It may also inspire new quantum algorithms and computational techniques, further advancing the field of quantum computation itself.

In summary, the exploration of semi-random formulae in superposition and the

potential discovery of true equations represent a bold and innovative application of quantum computation. This research holds the promise of significant advancements in mathematics and physics, offering new pathways for discovery and understanding in these fundamental sciences.

Background

Quantum computation is a field marked by profound advancements, largely driven by groundbreaking algorithms that have demonstrated the unique capabilities of quantum mechanics in solving complex problems. To understand the potential of applying quantum computation to discover mathematical truths, it is essential to review the key quantum algorithms and their applications in quantum mechanics, as well as the foundational concepts of superposition and quantum interference.

Review of Key Quantum Algorithms: Shor's and Grover's

Shor's Algorithm

Developed by Peter Shor in 1994, Shor's algorithm is one of the most celebrated achievements in quantum computation. It addresses the problem of integer

factorization, which involves decomposing a composite number into its prime factors. While classical algorithms for factorization, such as the General Number Field Sieve, are computationally expensive and scale exponentially with the size of the number, Shor's algorithm can factorize large integers in polynomial time. Shor's algorithm leverages quantum parallelism and the properties of the quantum Fourier transform to efficiently solve the factorization problem. The algorithm's ability to break RSA encryption, a widely used cryptographic scheme, highlights its transformative potential. The efficiency of Shor's algorithm stems from its capacity to process many possible factors simultaneously and interfere constructively to reveal the correct factors, showcasing the power of quantum computation.

Grover's Algorithm

Proposed by Lov Grover in 1996, Grover's algorithm provides a quadratic speedup for unstructured search problems. In a classical context, finding a specific item in an unsorted database of N entries requires, on average, N/2 queries.
Grover's algorithm reduces this to about √N queries, a significant improvement for large datasets. It applies the Grover iterator to amplify the probability amplitude of the correct solution while diminishing the amplitudes of incorrect ones. This process is analogous to repeatedly "shining light" on the correct answer until it becomes bright enough to be observed.

Grover's algorithm exemplifies the practical advantages of quantum search techniques and has potential applications in fields ranging from cryptography to database search.

Application of Quantum Computing to Quantum Mechanics Problems

Quantum computers are particularly well-suited to solving problems in quantum

mechanics, due to their ability to naturally simulate quantum systems. One prominent application is finding the ground state of a quantum system, which corresponds to the lowest energy configuration of the system's Hamiltonian. Classical algorithms often struggle with this task due to the exponential growth of the state space with the number of particles. Quantum algorithms, such as the Variational Quantum Eigensolver (VQE) and Quantum Phase Estimation (QPE), have been developed to address this challenge. VQE combines classical optimization techniques with quantum circuits to approximate the ground state

energy, while QPE accurately determines eigenvalues of a unitary operator, which can be used to find energy levels of a quantum system. These quantum algorithms have shown promise in various fields, including quantum chemistry, materials science, and condensed matter physics. By simulating quantum systems directly, quantum computers can provide insights into molecular structures, chemical reactions, and novel materials with unprecedented accuracy.

Discussion on Superposition and Quantum Interference

Superposition

Superposition is a fundamental principle of quantum mechanics, allowing a quantum system to exist in multiple states simultaneously. For a qubit, this means it can be in a state $|0\rangle$, $|1\rangle$, or any linear combination $\alpha|0\rangle+\beta|1\rangle$, where α and β are complex numbers such that $|\alpha|2 +|\beta|2$

```
=1. In quantum computation, superposition enables parallel
```

processing of vast amounts of information, exponentially increasing the computational power with each additional qubit.

Quantum Interference

Quantum interference occurs when the probability amplitudes of different quantum states combine. Constructive interference amplifies the probability of certain outcomes, while destructive interference diminishes others. This phenomenon is crucial for the operation of quantum algorithms, allowing them to highlight correct solutions and suppress incorrect ones. In algorithms like Shor's and Grover's, interference is used to amplify the correct answers. For example, in Grover's algorithm, the iterative application of the Grover iterator results in the constructive interference of the amplitude corresponding to the correct solution, making it more likely to be measured upon observation.

In the context of exploring semi-random formulae in superposition, quantum

interference could be harnessed to identify true mathematical equations. By designing algorithms that amplify the probability of valid formulae and suppress invalid ones, we can leverage quantum interference to potentially discover new mathematical truths.

Summary

The background provided here establishes the foundational concepts and key

algorithms that underpin the proposed exploration of semi-random formulae in superposition. Understanding Shor's and Grover's algorithms, the application of quantum computing to quantum mechanics problems, and the principles of superposition and quantum interference is crucial for appreciating the potential of this innovative approach. By building on these concepts, we aim to explore new frontiers in mathematical discovery through quantum computation.

The Concept of Semi-Random Formulae

The exploration of semi-random formulae in the context of quantum computation

represents a novel approach to discovering mathematical truths. This section delves into the definition and generation of semi-random formulae, the frameworks and rules governing their creation, and their mathematical and physical significance.

Definition and Generation of Semi-Random Formulae

Definition

Semi-random formulae are mathematical expressions generated by combining random elements with structured rules to create a diverse set of potential equations. Unlike purely random formulae, which are entirely arbitrary, semi-random formulae incorporate certain logical or syntactical constraints that make them plausible candidates for representing valid mathematical relationships or physical laws. The idea is to create a vast pool of formulae that, while generated with some degree of randomness, adhere to specific mathematical principles or structures. This increases the likelihood that some of these formulae may correspond to true or meaningful equations in mathematics or physics.

Generation

The generation of semi-random formulae can be achieved through a combination of random selection and deterministic rules. The process can be outlined as follows:

1. Element Pool: Define a pool of mathematical elements, such as variables,

constants, operators (e.g., +, -, *, /), functions (e.g., sin, cos, log, exp), and other symbols relevant to the domain of interest.

2. Structural Rules: Establish rules for how these elements can be combined to

form syntactically valid formulae. This could include rules for proper placement of operators, balanced parentheses, and correct use of functions.

3. Random Selection: Use random number generators or probabilistic methods to

select elements from the pool and combine them according to the structural rules. This can be done iteratively to produce a large number of formulae.

4. Filtering and Refinement: Optionally, apply additional filtering criteria to refine

the set of generated formulae, ensuring they meet certain desired properties or constraints.

Frameworks and Rules for Creating These Formulae

The creation of semi-random formulae requires a well-defined framework to ensure that the generated expressions are both diverse and meaningful. The framework should include the following components:

1. Grammar and Syntax Rules: Define a formal grammar that specifies the

allowable syntax for the formulae. This includes the order of operations, the hierarchy of functions, and the correct use of mathematical symbols.

2. Domain-Specific Constraints: Incorporate constraints specific to the domain of

interest. For example, in physics, ensure that the formulae are dimensionally consistent, meaning that the units on both sides of an equation match.

3. Randomization Techniques: Implement randomization techniques that allow for

the selection of elements and their combination. Techniques such as Monte Carlo methods, genetic algorithms, or other stochastic processes can be used to generate a wide variety of formulae.

4. Validation Criteria: Establish criteria for validating the generated formulae. This

might include checking for mathematical consistency, logical coherence, or alignment with known physical principles.

Mathematical and Physical Significance of These Formulae

Mathematical Significance

In mathematics, semi-random formulae represent a rich and unexplored landscape of potential relationships and structures. The significance of these formulae lies in their potential to:

1. Reveal New Patterns: By generating a wide variety of expressions, semi-random

formulae can help uncover new patterns and relationships that may not be immediately apparent through traditional methods.

2. Test Hypotheses: These formulae can serve as test cases for mathematical

hypotheses, providing a diverse set of examples against which theoretical propositions can be evaluated.

3. Inspire New Theorems: The discovery of interesting or unexpected relationships

among semi-random formulae could inspire the formulation of new mathematical theorems or conjectures.

Physical Significance

In physics, semi-random formulae have the potential to lead to new insights into the fundamental laws governing the natural world. Their significance includes:

1. Discovery of New Laws: By exploring a vast space of potential equations, we

may identify new mathematical representations of physical phenomena, leading to the discovery of new physical laws.

2. Modeling Complex Systems: Semi-random formulae can be used to model

complex systems, offering new ways to describe and understand the interactions and behaviors of such systems.

3. Expanding Theoretical Frameworks: The generation and analysis of these

formulae can help expand existing theoretical frameworks, providing new tools and perspectives for physicists to explore.

Summary

The concept of semi-random formulae represents an innovative approach to exploring mathematical and physical truths through quantum computation. By defining and

generating a vast array of potential formulae within a structured framework, we can leverage the principles of quantum mechanics to discover new relationships and insights. The mathematical and physical significance of these formulae lies in their potential to uncover new patterns, test hypotheses, inspire new theorems, and reveal new physical laws, ultimately advancing our understanding of the natural world.

Retrospective

The 2024 proposal should be read historically. Its important contribution was the question, not the specific quantum architecture. It assumed that mathematical discovery could be approached by generating semi-random formulae and then using quantum computation to favor valid ones. That already moved beyond ordinary theorem proving, but it left the human ontology almost untouched. The candidate language, operators, grammar, evaluation rules, and notion of a successful formula were all supplied before the quantum process began.

The later QRNG project emerged by attacking that weakness. If the goal is to test whether randomness can originate mathematical direction, then asking randomness to choose among prewritten statements gives it too little causal leverage. The statement space already embodies our habits. The object-first reversal asks for less semantic structure at the beginning: produce a finite object, compute intrinsic invariants, and postpone interpretation. The theorem, if one appears, should arise only after the object has been examined.

Chapter 2: Searching Is Not the Same as Originating

Adapted from Can Artificial Intelligence Discover Mathematics Before Humans Name It? (2026).

The next step was to distinguish calculation, deduction, conjecture, concept invention, and theory formation. Without that distinction, a fast search through a human-supplied language can be mistaken for origination.

1. From Mechanical Calculation to Machine-Generated Mathematics

The history of computational mathematics is often told as a steady progression from faster arithmetic to more powerful theorem proving. That account is incomplete. Calculation, deduction, conjecture formation, concept invention, and theory construction are distinct activities, and progress in one does not automatically imply progress in the others. A machine that evaluates a million expressions per second may contribute nothing to the discovery of a new mathematical object. A theorem prover may establish a difficult result without understanding why the theorem is interesting. A language model may generate plausible mathematical prose without producing a valid argument. The central question is therefore not whether computers can perform mathematical operations. They plainly can. The deeper question is whether they can participate in the formation of mathematics itself: selecting patterns, inventing useful distinctions, proposing laws, organizing results, and producing theories that were not already implicit in a human-specified target.

This distinction matters because mathematics is not merely a collection of true propositions. It is also a historical process by which objects, languages, axioms, and standards of importance are created. Groups, rings, manifolds, categories, and topological spaces were not simply waiting as named entities inside a fixed formal universe. They emerged when mathematicians recognized that disparate problems shared common structural features. The creative act often consisted not in proving one more theorem about familiar objects, but in deciding what the objects should be. Any serious account of machine-generated mathematics must therefore address not only automated reasoning within established theories but also the possibility of automated theory formation.

1.1 Calculation, Deduction, and Discovery

Calculation applies known operations to supplied inputs. Deduction derives consequences from supplied assumptions according to accepted rules. Discovery introduces something not previously specified as the immediate goal: a conjecture, concept, invariant, construction, classification, proof strategy, or theory. These activities overlap, but they should not be conflated. A computer algebra system may simplify an expression, factor a polynomial, integrate a function, or solve a system of equations. These achievements can be indispensable to research, yet the mathematical task has already been framed. The objects, operations, and desired output are known. The machine performs an extended calculation that would otherwise burden a human investigator.

Automated deduction begins at a higher logical level. Given axioms and a target proposition, a theorem prover searches for a derivation. The resulting proof may be surprising, shorter than

known proofs, or beyond unaided human reach. Nevertheless, the problem ordinarily arrives pre-formed. Humans determine the language, axioms, definitions, and theorem to be proved. The computer searches inside a mathematical world whose ontology has already been supplied. Discovery changes the specification itself. It may reveal that a previously unnoticed quantity is invariant, that two apparently separate domains share a structure, that a failed conjecture becomes true under a refined hypothesis, or that a family of examples should be recognized as a new kind of object. Discovery may therefore alter the space in which subsequent deductions occur. It does not merely traverse a theory. It may reorganize or enlarge the theory.

The strongest form of machine-generated mathematics would combine all three activities. It would calculate examples, deduce consequences, and use those results to revise its own concepts and questions. It would move recursively between data, language, law, and proof.

1.2 The Distinction Between Verifying and Originating Mathematics

Verification asks whether a proposed claim follows from stated assumptions. Origination asks where the claim, assumptions, and conceptual vocabulary came from. This distinction separates most traditional computer-assisted mathematics from the more ambitious project of artificial mathematical discovery. Proof assistants provide exceptionally strong verification. They require mathematical arguments to be expressed in a formal language and checked against precise logical rules. When a proof is accepted, the result has a level of syntactic and inferential reliability that ordinary prose rarely provides. Yet a proof assistant does not usually decide which theorem should be stated. It does not independently determine that a new definition captures an important phenomenon. It does not ordinarily recognize that a technical lemma should become the foundation of a new field.

Automated theorem provers move closer to origination by searching for proofs rather than merely checking human-written proof terms. Even so, most operate under a supplied goal. Their creativity is local: they originate a route through an established

logical space. They do not necessarily originate the space itself. A machine begins to originate mathematics when it participates in selecting what is worth proving, inventing the concepts needed to express it, and reorganizing results into a coherent body of theory. Such origination need not require consciousness, intention, or human-like understanding. It can be defined operationally. A system originates mathematical content when its output is not merely a direct execution of a fully specified human plan and when the output introduces a nontrivial structure, statement, or explanatory organization that was not explicitly encoded in advance.

This criterion remains difficult to apply. Every computational system inherits representations, algorithms, objectives, and evaluation rules from human designers. The question is therefore not whether human influence can be eliminated. It is whether the decisive mathematical content was supplied directly or emerged from a sufficiently open process.

1.3 Mathematical Creativity as a Computational Research Problem

Mathematical creativity is sometimes treated as too mysterious for formal investigation. That position prevents useful distinctions. Creativity can be decomposed into partially observable operations: generating candidates, transforming representations, detecting analogies, measuring novelty, searching for counterexamples, compressing regularities, introducing definitions, and evaluating explanatory power. Each component can become a computational research problem. Candidate generation can be studied through symbolic enumeration, stochastic search, program synthesis, or language models. Analogy can be approached through structural mappings between domains. Novelty can be estimated against databases of known results. Interestingness can be approximated through compression, unexpected implication, reuse across proofs, explanatory unification, or expert judgment. Counterexample search can be automated over finite models. Definitions can be evaluated by the theorems they enable and the complexity they remove.

No individual measure captures mathematical creativity. A rare statement may be useless. A highly compressive law may be trivial. A prolific definition may merely repackage known material. A system may rediscover a famous theorem without historical awareness and mistake rediscovery for novelty. The computational challenge is therefore not simply to maximize one score but to coordinate multiple imperfect criteria. Mathematical creativity also depends on scale. Automated systems can generate enormous numbers of true statements. This abundance creates a selection crisis. Human mathematicians do not value mathematics merely because it is correct. They seek results that explain, connect, simplify, generalize, or open productive lines of inquiry. The future of machine-generated mathematics will depend less on producing additional formal truths than on constructing mechanisms that distinguish fertile mathematics from combinatorial debris.

1.4 Five Levels of Machine Participation in Mathematics

Machine participation can be organized into five progressively stronger levels. The first level is verification. The machine checks arithmetic, symbolic transformations, proofs, or formal derivations supplied by humans. It functions as a reliability instrument.

The second level is proof discovery. The language, assumptions, and theorem are supplied, but the machine finds some or all of the argument. It functions as a search engine inside a defined theory. The third level is conjecture discovery. The machine examines examples, computations, or databases and proposes statements not explicitly requested in advance. It begins to influence the research agenda, although the object domain and descriptive vocabulary are usually predetermined. The fourth level is concept invention. The machine introduces new predicates, invariants, classifications, constructions, or representations that make previously inaccessible regularities expressible. It changes not only what can be proved but what can be stated.

The fifth level is theory or ontology formation. The machine organizes concepts and laws into a coherent mathematical world, potentially identifying its fundamental objects, equivalences, transformations, and explanatory principles. At this level, the system does not merely solve problems within a domain. It participates in determining what the domain is. These levels are not perfectly separated. A proof-discovery system may introduce a useful lemma that functions as a new concept. A conjecture generator may reorganize a field by revealing an unexpected invariant. A theory-forming system still depends on verification and deduction. The hierarchy is therefore functional rather than absolute. Its purpose is to prevent impressive achievements at one level from being mistaken for achievements at another.

1.5 The Unresolved Problem of Autonomous Theory Formation

Autonomous theory formation remains the least developed level. Existing systems can generate proofs, equations, programs, conjectures, and classifications. Some can modify their own search strategies or invent derived concepts. Yet most begin with a substantial inherited ontology. They are told that the relevant objects are numbers, graphs, groups, geometric configurations, programs, or formal propositions. They are given predicates, constructors, axioms, evaluators, training corpora, or benchmark goals that already encode much of the mathematical world they will explore. The unresolved problem is whether a computational system can begin beneath this level of specification. Can it encounter minimally interpreted structure and discover stable objects, useful equivalences, candidate laws, and organizing principles without being told which familiar theory to reconstruct? Can it distinguish genuine structure from accidental regularity? Can it form multiple competing theories, compare their explanatory productivity, reject weak

generalizations, and preserve the strongest survivors? Can it recognize when a new vocabulary is required?

No system begins from literal nothing. A carrier set, representation scheme, computational substrate, or observation policy must always be supplied. The meaningful question is how little mathematical interpretation must be imposed before theory formation begins. The boundary between necessary infrastructure and covertly supplied

mathematics is central. Algebraic Abiogenesis enters at this boundary. It does not attempt to eliminate all prior structure. Instead, it reduces the initial commitment to finite carriers, anonymous operations, generated expressions, and exact evaluation. It then asks whether identities, theory families, and eventually recognizable algebraic objects can emerge from repeated behavioral equivalence. The project therefore belongs to a longer history of automated discovery while advancing a more specific challenge: whether mathematics can be induced before its objects and axioms have been named.

Chapter 3: From Randomness to Information

Adapted from Pure Information from Quantum Randomness (2025). Metaphysical possibilities are retained as historical context; the later theorem does not depend on them.

Two conceptual changes followed. First, the project stopped treating randomness as a message that necessarily contained hidden semantic structure. A random string need not conceal a code in order to be mathematically productive. Second, the project stopped asking the QRNG to supply an answer. Its role would be generative rather than oracular: determine a starting object that no one selected for its mathematical attractiveness. From that point onward, deterministic computation, proof, counterexample, and literature review would carry the burden.

This separation removes a metaphysical dependency. The finite theorem in the present repository is true regardless of how one interprets quantum measurement. What matters experimentally is that the seed was intended to be supplied independently of the later mathematical anomaly. What matters mathematically is that once the conjecture was formulated, it was settled without appeal to randomness.

1. Introduction: From Quantum Gadgets to Informational Ontology

Quantum random number generators are usually introduced in engineering terms: black boxes that transform quantum indeterminacy into high-quality randomness for cryptography, simulations, and secure protocols. They sit quietly in racks or on optical benches, converting photon detections or spin measurements into streams of bits that feed into larger systems. In that role they are unremarkable: just another component in the modern information stack. In this paper, we treat them instead as conceptual probes. Every time a QRNG emits a bit, it realizes a particular kind of event: a well-characterized quantum state is prepared, a measurement in a chosen basis is performed, and a classical outcome is irreversibly recorded.

From the standpoint of standard quantum mechanics, the individual outcome is fundamentally unpredictable, even if the state and the measurement are fully known. From the standpoint of information theory, long sequences of such outcomes behave as if they were algorithmically incompressible. These two facts already suggest that QRNG outputs occupy a special place in any informational interpretation of physics. If the universe is, in some sense, an information-processing structure, then the bits returned by a QRNG are among the simplest, most controllable examples of "raw information events" we can generate on demand. The central move of this paper is to ask whether we should take this more seriously: to see QRNGs not just as quantum gadgets but as small, repeatable interfaces between the deep structure of reality and the classical informational layer in which agents live and reason.

Within that shift, we will frame QRNG bits as candidates for "pure information," and articulate what that phrase could mean. We will introduce a simple four-layer stack, Sigma -> Q -> B -> P, and connect it with a logostic perspective in which agent

trajectories q(t) seek alignment with a generative manifold tau(t). The introduction lays out these motivations and definitions, setting up the more detailed analysis that follows.

1.1 The practical role of QRNGs in today's technology

In contemporary practice, QRNGs are sold and deployed as specialized components whose purpose is to provide random bits with stronger guarantees than classical pseudorandom generators. In cryptographic systems, they seed or supplement key generation, ensuring that secret keys cannot be reconstructed by an adversary who knows the algorithm but not the particular random choices. In high-stakes simulations, such as Monte Carlo calculations for finance, physics, or engineering, they serve as a source of nondeterministic variability that is not tied to the internal state of a software generator.

Typical QRNG implementations rely on simple, well-understood quantum processes. A photon can be sent to a balanced beam splitter and detected at one of two outputs, mapping detector A to bit 0 and detector B to bit 1. Alternatively, a two-level system can be prepared in a superposition and measured in a fixed basis, with outcomes associated to bits. In all such cases, the core idea is the same: prepare a state whose Born-rule probabilities are known, and then let quantum mechanics produce a discrete, classical outcome. Engineers worry about bias, correlations, hardware failures, and classical side channels. They design post-processing schemes to remove residual imperfections, and they certify devices by running large batteries of statistical tests on output streams. Within that context, QRNGs are evaluated by how closely their outputs approximate the behaviour of an ideal random source: equal frequencies of 0 and 1, absence of detectable patterns in subsequences, and stability over time.

This pragmatic framing is important. It anchors QRNGs in real devices, real measurements, and real use-cases. But it leaves a conceptual gap. The devices are treated as boxes that "produce randomness," while the deeper question-what kind of event a quantum random bit actually is in an informational universe-remains largely bracketed. This paper steps into that gap, using the familiar practical role of QRNGs as a starting point for a more ontological inquiry.

1.2 From unpredictability to information: why random bits matter conceptually

Random bits are not only useful; they are conceptually revealing. In Shannon's framework, a single fair bit has maximal entropy among all two-outcome variables: it carries one bit of information precisely because, before observation, the two outcomes are equally likely and no deterministic prediction is possible. A long sequence of such bits can, in principle, encode any message; it is the most flexible possible raw material for description. Algorithmic information theory sharpens this point. A finite bit string is considered algorithmically random if there exists no substantially shorter program that can generate it on a universal computing device. For such a string, the shortest effective description is essentially the string itself. In that sense, purely random sequences are "pure information" in the minimal sense: they cannot be compressed without loss, and their content is not reducible to a simpler underlying pattern.

Random bits matter conceptually because they show us what it means for information to be irreducible. When we see patterns, we can describe them succinctly; when we do not, we either lack insight or the pattern genuinely is not there. Randomness forces us to confront this distinction. It also touches directly on the relationship between law and contingency. Physical laws typically constrain probabilities and correlations, but they do not specify individual

outcomes. Random bits sit precisely at that interface between lawful structure and particular realized histories. For an informational ontology-for any worldview that treats information as fundamental or near-fundamental-these bits are not mere noise. They represent the points at which the universe "chooses" between multiple possibilities, or in more neutral language, the points at which one of several allowed descriptions becomes actual. Understanding what kind of information these choices embody, and how they relate to deeper layers of structure, is therefore a central task. QRNGs provide a clean laboratory where such questions can be posed.

1.3 Framing the central question: are QRNG bits pure information?

The central question of this paper is whether the bits produced by QRNGs should be regarded as "pure information," and if so, in what sense. At a minimal level, QRNG outputs look like good candidates. When the device is properly designed, the measured bitstream passes stringent statistical tests for independence and uniformity. No short classical description predicts the exact sequence of outcomes. Algorithmically, the sequence behaves as if it were incompressible. However, calling QRNG bits "pure information" can mean at least two different things. In the minimal, information-theoretic sense, it means only that there is no effective compression and no exploitable pattern: the bits are maximally unpredictable given public information about the device and its operation. In this sense, QRNG output is pure because it contains no structure that can be used to reduce its description length. It is information without pattern.

In a stronger, metaphysical sense, "pure information" could mean that QRNG bits are direct manifestations of a deeper informational substrate, rather than arbitrary noise. On this reading, each bit is a one-bit projection of a vastly higher-dimensional state, carrying traces of a hidden structure that we lack the language to express. What appears random from our perspective might be lawful and patterned at a more fundamental level, with the QRNG serving as a lossy interface. The paper does not claim to settle this stronger question definitively. Instead, it explores how one might systematically articulate these two notions of purity and examine QRNG outputs in both lights. The aim is to move from a vague intuition-"these bits feel like pure information"to a more precise conceptual landscape in which that intuition can be assessed, refined, and connected to broader questions about the informational nature of physical reality.

1.4 Overview of the Sigma -> Q -> B -> P stack and the logostic perspective

To make sense of QRNG events in an informational ontology, it is useful to distinguish several layers of description. We will work with a four-layer stack: Sigma, Q, B, and P. Sigma denotes a hypothesized deep informational substrate: the level at which whatever fundamental order or Logos underlies the universe is expressed. This layer is not described directly by current physical theories, but it is the natural home for any proposal that treats information as more primitive than spacetime and fields. The Q layer corresponds to the standard quantum formalism. Here we speak of Hilbert spaces, quantum states, amplitudes, and unitary evolution. Quantum theory can be seen as a structured interface between Sigma and the more classical layers, encoding how deep informational constraints manifest as probabilities and correlations among measurement outcomes. In practice, QRNGs operate primarily at this layer: they prepare controlled quantum states and subject them to well-defined measurements.

The B layer is the bit layer: the realm of discrete classical outcomes. When a detector fires, or a register flips between 0 and 1, an event in B has occurred. QRNGs are designed precisely to yield such B-level events, with each quantum measurement mapped to a 0 or a 1. Finally, the P layer is the physical-event layer in everyday terms: recorded data, logged files, photons counted, voltages measured. It is the level at which experimental runs, device specifications, and engineering workflows are described. Against this layered background, we introduce a logostic perspective in which agents are described by trajectories q(t) through state spaces, and in which there exists a generative manifold tau(t) that encodes time-dependent "living truth" or structurally privileged configurations. Logostics is concerned with how q(t) moves toward or away from tau(t), and with the informational forces that shape that motion.

The connection to QRNGs is this: each QRNG bit can be seen as a small, localized event in the B layer, constrained by the Q layer and ultimately rooted in Sigma. Agents living at the P and B layers can harness these events, incorporating them into their decision processes, models, and explorations. If Sigma is genuinely informational, and if tau(t) is a meaningful projection of that deep structure, then QRNG bits are not merely convenient randomness, but elemental points at which Sigma's constraints touch the agent's informational world. The rest of the paper elaborates this picture, contrasting QRNG randomness with pseudorandomness and exploring its implications for cosmology, life, and alignment within the logostic framework.

2. Randomness, Information, and Algorithmic Complexity

Before asking whether quantum random number generator outputs qualify as "pure

information," it is necessary to clarify what randomness and information mean in more technical terms. Two complementary frameworks are especially relevant. The first is Shannon's probabilistic theory of information, which quantifies average surprise over repeated draws from a distribution. The second is algorithmic information theory, which focuses on the descriptive complexity of individual sequences. Together, they provide both an ensemble view and a singlesequence view of what it means for data to be random and informative. In this section, we briefly review Shannon's notion of entropy and the meaning of one bit of information, then introduce algorithmic randomness and Kolmogorov complexity as measures of incompressibility. We then contrast practical statistical testing of random number generators with the stronger, though idealized, notion of being algorithmically random. Finally, we consider under what conditions it is reasonable, in practice, to treat a finite sequence as "pure information," even when strict theoretical proofs of randomness are impossible. This conceptual

groundwork will serve as the lens through which QRNG outputs are examined in subsequent sections.

2.1 Shannon information, entropy, and the meaning of "1 bit"

In Shannon's information theory, information is defined relative to uncertainty. Consider a discrete random variable X that can take values x with probabilities p(x). The Shannon entropy

H(X) is defined as

```
H(X) = - sum_x p(x) log_2 p(x).
```

Entropy measures the expected surprise associated with observing an outcome of X. When one outcome is certain (p(x) = 1 for some x), there is no surprise and H(X) = 0. When the distribution is more spread out, entropy increases, reaching its maximum when all outcomes are equally likely. The basic unit in this framework is the bit. A random variable with two outcomes, each with

```
probability 1/2, has entropy H(X) = 1 bit. This means that, on average, each realization of
X
```

conveys one bit of information: it selects one of two equally plausible possibilities. More generally, entropy in bits can be interpreted as the minimum average number of yes/no questions needed to learn the outcome, assuming we can choose the questions optimally. This links information directly to choice among alternatives. "One bit" is not a mysterious substance; it is simply the informational cost of resolving a binary uncertainty. A source that reliably outputs fair bits is, in Shannon's sense, providing the maximum possible information per

binary symbol. Any bias or predictability reduces the entropy and therefore the information per symbol. For streams of symbols, Shannon's theory addresses expected properties over many realizations. A long sequence of independent fair bits has an

entropy rate of 1 bit per symbol. This does not directly tell us whether any particular finite sequence is "random," but it does characterize the ideal behaviour of a source that is as unpredictable as possible under a given alphabet and probability model.

2.2 Algorithmic randomness and Kolmogorov complexity

Shannon's theory quantifies average information given a known probability distribution, but it does not directly address the question: is this specific finite sequence random? To tackle that, algorithmic information theory introduces Kolmogorov complexity. For a finite binary string s, the Kolmogorov complexity K(s) is defined as the length, in bits, of the shortest program on a fixed universal computing device that outputs s and then halts. Intuitively, K(s) measures the shortest effective description of s. A highly structured sequence, such as 01010101 repeated many times, has low Kolmogorov complexity because a very short program can generate it (for example: "print 01, n times"). By contrast, a sequence with no discernible pattern will typically require a program that is essentially "print this exact sequence," whose length is about the same as the sequence itself.

A sequence is called algorithmically random, or incompressible, if its Kolmogorov complexity is close to its length. For an n-bit string s, this means K(s) is at least n minus a modest constant. Such a string has no significantly shorter description; there is no compact explanation of its pattern because, in a precise sense, it has no exploitable pattern. This notion of randomness is stronger than statistical unpredictability in a specific test. A string could conceivably pass many statistical tests yet still be generated by a relatively short algorithm, in which case its Kolmogorov complexity is low. Algorithmic randomness demands that the string escape all such algorithmic pullbacks; it looks patternless even when we allow arbitrary programs to search for structure.

In practice, Kolmogorov complexity is not computable exactly, but the concept provides a powerful idealization. It formalizes the idea that truly random sequences are those for which "the best you can do is just restate them," with no shorter explanatory handle.

2.3 Statistical tests versus deeper notions of incompressibility

When evaluating random number generators, including QRNGs, practitioners rely on statistical test suites rather than direct assessments of Kolmogorov complexity. These tests probe various properties of a sequence: the frequency of zeros and ones, the distribution of runs of the same bit, correlations between positions, behaviour of overlapping patterns, and more. If the generator is ideal, the observed statistics should fall within ranges predicted by the assumed random model. Passing a battery of tests provides evidence that the sequence does not exhibit certain specific kinds of structure. For example, it might show no significant bias, no obvious periodicity, and no detectable correlations of low order. However, this is always relative to the set of tests considered. A sequence might pass all the tests in a given suite and yet still be compressible by some cleverly constructed algorithm that exploits a subtle pattern not covered by those tests.

Algorithmic randomness, by contrast, demands robustness against all effective tests and compression schemes, not just a finite catalog. In the limit of infinite sequences, notions such as Martin-Lof randomness formalize this idea: a sequence is random if it

passes all effectively specifiable tests that would be failed by only a small measure of sequences. Kolmogorov complexity and these test-based notions are closely related in the infinite limit, but for finite sequences used in practical systems, they remain ideal reference points rather than checkable conditions. This gap between practice and ideality is important. It implies that for any physically realizable testing regime, one can never prove that a given finite sequence is truly incompressible. One can only say that, relative to the tests applied and the models considered, no structure has been detected. The possibility of undiscovered structure always remains in principle, even if it becomes increasingly implausible as more refined tests are passed.

2.4 When a sequence becomes "pure information" in practice

Given these limitations, what does it mean in practice to call a sequence "pure information"? Strictly speaking, the only sequences that are provably incompressible are those that are defined so by construction in mathematical arguments; any exhibited finite sequence might have undiscovered structure. Yet in real systems we are forced to operate under finite resources and bounded model classes. In that context, purity becomes a relative, operational notion rather than an absolute one. A sequence can be treated as pure information when, within a given modeling and testing framework, three conditions are met. First, it passes all relevant statistical tests designed to

detect biases, correlations, or simple patterns. Second, attempts to compress it using a broad class of practical algorithms fail to produce significant reductions in length. Third, there is no known physical or algorithmic mechanism that would generate the observed sequence with a short description, given the publicly known setup. Under these conditions, the most economical stance is to regard the sequence as informationally irreducible relative to our current knowledge and tools. Its shortest effective description, for all practical purposes, is the sequence itself. Any further attempt to find hidden structure yields diminishing returns. At that point, engineers and theorists are justified in treating the sequence as a realization of an ideal random source, even though absolute guarantees are unavailable.

In the context of this paper, this is what is meant by "pure information" in the modest sense: a bitstream that, within our best available tests and models, behaves as if it were algorithmically random. For QRNG outputs, this practical notion of purity is already significant. It positions the QRNG not merely as a provider of convenient noise, but as a device that, as far as we can tell, injects genuinely new descriptive content into the classical informational world. Later sections will examine how, on a stronger metaphysical reading, such sequences might also be interpreted as one-bit shadows of deeper structures, linking this practical purity to an ontological role in an informational cosmos.

3. A Four-Layer Stack: Σ, Q, B, and P

To talk coherently about quantum randomness and its informational significance, it helps to separate different layers at which we describe the world. Quantum theory itself blurs some of these boundaries, and philosophical discussions often jump freely between them, but for our purposes a simple four-layer stack is useful: Σ, Q, B, and P. Σ is a hypothesized informational substrate or deep order; Q is the familiar quantum formalism of states and amplitudes; B is the layer of discrete classical bits; and P is the layer of macroscopic physical events and records. This stack is not offered as a new physical theory, but as a conceptual scaffold. It lets us ask precise questions: where exactly do QRNGs "live"? At which interfaces does randomness enter?

How might an informational or logostic ontology be mapped onto the layers that physics and engineering already use? By distinguishing these levels clearly, we can better understand what it means for QRNG bits to be both constrained by deep structure and irreducible at the classical level.

3.1 Defining Σ: informational substrate, Logos, or deep order

The Σ layer is the most speculative and least directly observable part of the stack. It is meant to denote whatever deep structure underlies both quantum theory and classical reality if one takes information or Logos-like order to be fundamental. Various traditions have proposed candidates for such a level: mathematical Platonism, informational cosmologies, and theological conceptions of Logos all gesture toward an underlying domain in which coherence, constraint, and generative order are primary. In the present framework, Σ is not defined by specific equations or models. Instead, it is characterized by its role. It is the level at which the possibilities encoded in the universe are structured before they are expressed as quantum states or classical histories. One can imagine Σ as a vast space of constraints, symmetries, and generative rules that determine which patterns can exist at higher layers, even if those patterns appear random when seen from a limited vantage.

Importantly, Σ is not simply "the set of all physical laws" in a conventional sense. It may include those laws, but also the deeper informational regularities that make those laws possible, such as the existence of stable particles, the dimensionality of space, or the capacity for computation. If one speaks of a Logos, Σ is where that Logos operates. If one favors a strictly secular vocabulary, Σ can still serve as a placeholder for the unknown informational architecture that makes quantum and classical descriptions cohere. This way of speaking is intentionally modest. It does not commit us to a detailed theory of Σ, only to the idea that it is meaningful to consider a level at which the universe's capacity for regularity and novelty is grounded. QRNGs then become interesting because they may reveal how this deep order shows up at the level of individual bits.

3.2 Q as the qubit-layer: Hilbert spaces, amplitudes, and unitary evolution

The Q layer is where standard quantum mechanics lives. Here, systems are described by vectors in Hilbert spaces, or more generally by density operators. Dynamics is given by unitary evolution, interrupted by measurement interactions that correlate system states with apparatus states. Probabilities for outcomes are computed via the Born rule, which assigns weights to different components of a state in a chosen basis. Q is the domain of qubits, superpositions, entanglement, and interference. In a QRNG based on a beam splitter, for example, a single photon is prepared in a state that can be represented as a superposition of "reflected" and "transmitted." At this level, nothing has yet become a definite

classical outcome. The state is a compact, highly structured object encoding the amplitudes for different possibilities. From the point of view of the four-layer stack, Q is an interface between Σ and more classical layers. Whatever deep informational structure Σ embodies, it is expressed at Q as the particular Hilbert spaces that exist, the allowed Hamiltonians, and the families of states that can be prepared and evolved. Quantum theory is thus a kind of midlevel language: more concrete than an abstract Logos, but more general and non-classical than the bit-level descriptions used by digital systems. Crucially, Q is still reversible at the level of pure dynamics. Unitary evolution preserves information in a strong sense; no bits are created or destroyed, only rearranged in the space of amplitudes. The apparent introduction of randomness occurs only when we consider the transition from Q to more classical descriptions, a point that becomes central when we examine the bit layer and the role of measurement.

3.3 B as the bit-layer: measurement outcomes as classical yes/no events

The B layer is the realm of bits: discrete, classical yes/no outcomes. When a QRNG reports "0" or "1," it is delivering an event at this level. B is not concerned with amplitudes or superpositions; it is concerned with definite alternatives that can be encoded in classical registers, transmitted over channels, and reasoned about in ordinary Boolean terms. In physical implementations, the transition from Q to B occurs through measurement interactions. A quantum system prepared at Q couples to a macroscopic apparatus, and as a result, one of several possible pointer positions becomes effectively definite. At the conceptual level, this is where the Born-rule probabilities at Q are turned into actual frequencies of bits at B. In engineering practice, this is also where device imperfections, detector efficiencies, and noise enter the picture.

The interest of B for informational ontology lies in the fact that it is the level at which "outcomes" exist in the sense relevant to Shannon and algorithmic information theory. A bitstring at B can be compressed, tested, and used to seed cryptographic keys. Each bit is a choice among alternatives; each string is a particular realized pattern. If the Q layer describes the weighted possibility space, the B layer records the realized path taken through that space in terms of binary decisions. In a QRNG, every measurement is deliberately engineered to map a Q-level superposition onto a B-level outcome. Detector A is defined to correspond to bit 0, detector B to bit 1. In ideal operation, the bitstream emerging at B is maximally unpredictable given public knowledge of

the Q-level preparation. This is what makes QRNGs interesting: they are devices whose entire purpose is to repeatedly enact the Q-to-B transition under controlled conditions.

3.4 P as physical events: detectors, records, and macroscopic facts

The P layer is the domain of physical events as described in everyday experimental and engineering language. Here we speak of detectors clicking, voltages crossing thresholds, files being written to disk, and timestamps being logged. P is not distinct from B in a strict physical sense-bits must always be represented in physical media-but it is conceptually useful to separate "the abstract bit pattern" from the concrete details of how that pattern is instantiated. In practice, P is where QRNG specifications live. An engineer might describe the incident photon flux, the detector response time, the temperature ranges, and the data rates. They might note how the device is powered, what interfaces it provides, and how its output is post-processed to remove residual bias. These are all P-level descriptions: they talk about the macroscopic, operational reality surrounding the bitstream.

P is also where agents encounter QRNG output. A user does not experience abstract bits in isolation; they see numeric strings on a screen, files stored in directories, or keys loaded into cryptographic modules. The entire apparatus of experimental control, error correction, and certification is conducted at the P layer. When we ask whether a QRNG is "working properly," we are asking whether the physical events at P are consistent with the expectations we have for B-level randomness under Q-level modeling. This separation matters because the same abstract bitstream could, in principle, be instantiated by many different P-level mechanisms. The distinction allows us to ask: given that we observe certain P-level regularities, and that we interpret them as evidence about B and Q, what might they imply about Σ? Without confusing the concrete hardware description with the informational content, we can map how evidence flows up and down the stack.

3.5 QRNGs as explicit interfaces between Q and B

With these layers in place, we can situate QRNGs precisely. A QRNG is an engineered interface between Q and B, wrapped in a P-level device. It takes carefully prepared quantum states in Q, subjects them to controlled measurement operations, and produces B-level bits that are recorded and manipulated at P. Its purpose is to repeatedly realize the Q-to-B transition in a way that maximizes unpredictability at B while remaining transparent and testable at P.

Viewed this way, QRNGs are not merely gadgets but small, repeatable experiments in which a deep informational structure (Σ) is filtered through the quantum formalism (Q) and expressed as classical outcomes (B) in a physically accessible context (P). Every emitted bit is the endpoint of this cascade: constrained by Σ, shaped by Q, discretized at B, and manifested at P. QRNGs thus provide a uniquely clean setting for studying how randomness and information emerge as we move from abstract possibilities to concrete bits. This perspective also sharpens the contrast with pseudorandom number generators. A pseudorandom generator operates entirely at the B and P layers: it takes existing bits, transforms them with deterministic algorithms, and produces outputs that look random

but are fully determined by their initial state. No Q-level superpositions or Σ-level constraints enter into the production of new bits; the apparent randomness is entirely classical.

By contrast, a QRNG uses Q to inject new bits into B that, as far as we can tell, have no shorter description than themselves. If one adopts an informational or logostic ontology, QRNGs are therefore more than sources of convenient noise. They are windows-however narrow-into the way Σ, through Q, produces particular classical histories. The rest of this paper explores what it might mean to treat those histories, and especially the QRNG bitstreams, as instances of "pure information" in both a modest, operational sense and a richer, metaphysical one.

4. Pseudorandomness vs Quantum Randomness

With the four-layer stack in place, we can now sharpen the distinction between classical pseudorandomness and quantum randomness. Both can produce bitstreams that pass extensive statistical tests. Both can be used for cryptography, simulations, and protocol design. Yet from the standpoint of information and ontology, they occupy very different roles. Pseudorandom generators live entirely within the bit and physical layers. They transform existing classical information into new-looking patterns without ever leaving the realm of deterministic computation. Quantum randomness, by contrast, relies on the qubit layer and, in this framework, ultimately on the Σ layer: it injects bits that are not reducible to prior B-level patterns. This section develops that contrast by examining classical pseudorandom generators,

```
focusing on SHA-256 as a paradigmatic B → B transform, and then contrasting this with
quantum randomness understood as Σ/Q → B injection. The goal is to show why, even if both
```

sources pass similar tests, QRNG outputs are qualitatively different in how they relate to deep informational structure.

4.1 Classical pseudorandom generators: structure disguised as noise

Classical pseudorandom number generators (PRNGs) are deterministic algorithms that take a finite seed and produce a long sequence of bits that appears random to observers lacking knowledge of the seed. Linear congruential generators, Mersenne Twister, and cryptographically secure PRNGs all share this basic form: they are functions from a short initial description to a much longer output that passes many statistical tests. The key point is that PRNG output is compressible in principle. Given the algorithm and the seed, the entire sequence can be generated by a relatively short program. The Kolmogorov complexity of the output string is therefore bounded by the length of the seed plus the algorithmic overhead, which is usually small. Even if the output looks patternless, there is a compact explanation of it in terms of the generator's internal state and update rules.

From an information-theoretic perspective, PRNGs do not create new information; they only reexpress existing information in a more complicated form. The entropy of the output is at most the entropy of the seed, because the mapping is deterministic. In practice, this is acceptable for many purposes. If the seed is truly unpredictable to an adversary, then the output will be unpredictable as well, even if it contains no additional

entropy. But for an ontology that takes information as fundamental, this distinction matters: PRNGs recycle; they do not mint. Cryptographically secure PRNGs strengthen this point. They are designed so that, without the seed, no efficient algorithm can distinguish their output from true randomness. Yet the existence of a short generating program remains. The apparent randomness is structure disguised as noise, not an absence of structure. Under the hood, the generator is marching deterministically through a huge but pre-defined state space.

4.2 SHA-256 as a B → B transform: determinism with a short description

SHA-256 provides a particularly clear example of a deterministic bit-to-bit transform that produces random-looking output. Given an input bitstring, the hash function computes a 256bit digest through a fixed sequence of logical operations, modular additions, and bitwise permutations. For any particular input, the output is completely determined; hashing the same input twice gives the same digest. From the B-layer perspective, SHA-256 is a map from one classical bitstring to another. It does not depend on any quantum state or external source of entropy. Its behaviour can be fully specified by a relatively short algorithmic description: the standard itself plus the particular input. The Kolmogorov complexity of the output string, given the input and the algorithm, is

therefore low. A short program that contains the SHA-256 procedure and the input can generate the digest exactly. Despite this, SHA-256 outputs behave statistically like random bitstrings when the inputs are not adversarially chosen and the function is treated as a black box. Each output bit is close to unbiased; small changes in the input produce large, apparently uncorrelated changes in the output; and aggregated outputs over many inputs pass standard randomness tests. Cryptographic security assumptions go further, treating SHA-256 as a pseudorandom function: no efficient adversary should be able to distinguish its outputs from truly random strings without knowing the inputs.

Thus, SHA-256 realizes the ideal of pseudorandomness: structured determinism that is empirically indistinguishable from randomness for computationally bounded observers. In the

```
present framework, however, this remains a purely B → B operation. All the "randomness" is
```

the product of a fixed classical transform. No new descriptive content enters from outside the classical bit world.

4.3 Quantum randomness as Σ/Q → B injection: new description content

Quantum randomness, as realized in a QRNG, has a different character. The device begins with a quantum state at the Q layer: for example, a photon in a superposition of two paths, or a twolevel system prepared in a balanced superposition of basis states. This state is then subjected to a measurement, producing a classical outcome that is assigned to bit 0 or bit 1 at the B layer. In standard quantum mechanics, the probability of each outcome is determined by the squared amplitudes of the state in the measurement basis, via the Born rule. Yet the specific outcomewhether the next bit will be 0 or 1-is not determined by any classical data accessible in advance. Even a complete description of the Q-level state, together with the measurement setup, only yields probabilities. The actual bit realized at B is chosen in a way that cannot be predicted better than those probabilities allow.

```
Within the four-layer stack, this can be viewed as a Σ/Q → B injection of new description
```

content. The Q layer encodes lawful structure in its amplitudes and dynamics; Σ represents the deeper informational constraints that make that structure possible. When a measurement occurs, something that was previously expressed as a superposed possibility distribution becomes a definite bit. That definite bit, seen as part of a longer sequence, typically has no shorter B-level description than itself, given publicly known information about the device. From an algorithmic standpoint, if we treat the QRNG's physical design, control software, and environmental conditions as fixed and known, the only remaining source of variation in the

output is the measurement outcome itself. There is no short classical program that can reproduce the exact sequence in advance, because no such program exists even in principle under the usual interpretation of quantum theory. Each bit thus represents a genuine addition to the classical description of the world: a new choice among possibilities that cannot be reduced to prior B-level data. This is what is meant here by "new description content." The QRNG is not merely scrambling existing bits according to a deterministic rule; it is drawing on a Q-level process that, in turn, reflects deeper Σ-level constraints. The resulting B-level sequence extends the classical history with bits that are not functions of what came before.

4.4 Why QRNG output is qualitatively different from cryptographic pseudorandomness

From an engineering standpoint, QRNGs and cryptographic PRNGs can sometimes be

interchanged. Both can provide unpredictable bits for keys and simulations. Yet at the level of informational ontology, the difference is stark. A cryptographic PRNG stretches an initial stock of entropy into a longer stream, preserving but not increasing the total amount of information. Its outputs, however convincing, are shadows cast by a small seed and a known algorithm. QRNG outputs, by contrast, are not compressible to a small classical seed plus a deterministic program within the usual quantum framework. The best available description of a long QRNG sequence, given the device specification, is simply the sequence itself. Even if one postulates a deeper Σ-level determinism, that determinism is not expressible in B-level terms in any way that would allow classical prediction or compression. For observers confined to B and P, QRNG bits are informationally irreducible in a sense that PRNG bits are not.

Another way to put it is that cryptographic pseudorandomness is epistemic: the unpredictability arises from ignorance of the seed. Quantum randomness, as implemented in QRNGs, is ontic at the B-level: even complete knowledge of the Q-level state does not permit better-thanprobabilistic prediction of individual outcomes. The apparent randomness is not merely a placeholder for hidden classical variables in standard interpretations; it is a reflection of how Σlevel constraints are expressed through Q-level structure into B-level events.

```
Finally, from the perspective of logostics and the q(t) → τ(t) framework, this difference
matters
```

because only QRNG-like processes genuinely extend the classical description space along an agent's trajectory. PRNG-based randomness reuses existing information; QRNG-based randomness adds bits that, for all practical purposes, did not exist in the classical description beforehand. When agents incorporate QRNG bits into their decisions, they are therefore coupling themselves more directly to the Σ/Q interface than when they rely solely on deterministic algorithms.

For these reasons, QRNG output is qualitatively different from cryptographic

```
pseudorandomness. Both may look random to our tests, but only QRNGs realize the full Σ → Q
→ B → P cascade that makes them suitable candidates for "pure information" in the modest
```

operational sense, and for more ambitious metaphysical readings that see them as one-bit shadows of a deeper informational order.

Part II - Mathematics Before Names

Chapter 4: From Marks to Terms

Adapted from The Virgin Oracle, Chapter 2 (2026).

A distinction does not yet constitute a language. A machine may recognize that one state differs from another without possessing any durable way to describe either state. To build a mathematical world, the virgin oracle must acquire a system of marks that can be stored, repeated, combined, and interpreted. Let Σ denote a finite alphabet. At the beginning, its characters need not carry mathematical meanings. They are simply distinguishable marks:

```
Σ = {A, B, x, y, 0, 1, =, ·, (, )}
```

The exact inventory is not fundamental. Another intelligence might use electrical states, colored nodes, geometric shapes, or integer codes. What matters is that each primitive sign can be distinguished from the others and that finite sequences of signs can be stored without ambiguity. From the alphabet, the oracle can form strings. The set of all strings of length n is: Σ^n The set of all finite strings is:

```
Σ* = ⋃ₙ≥0 Σⁿ
```

This construction is enormous even when the alphabet is small. If Σ contains k characters, there are k^n strings of length n. Most of those strings will carry no coherent interpretation. They are merely possible arrangements of marks. The oracle therefore faces a problem that precedes mathematics: Which strings count as meaningful expressions? The alphabet alone cannot answer. Meaning requires structure imposed on sequences. Suppose the oracle possesses three variable symbols: x, y, z and one symbol intended to represent a binary operation: ·

The raw string

x · y can be interpreted as applying the operation to x and y. But strings such as · · x or x y · · z do not yet possess a clear interpretation under the same convention. The oracle needs a grammar. A minimal grammar for binary terms can be stated recursively: Every variable is a term. If s and t are terms, then (s · t) is a term. This definition does more than select acceptable strings. It gives every valid expression an internal construction history.

The expression

((x · y) · z) was formed by first combining x with y and then combining the result with z.

The expression

(x · (y · z)) was formed by first combining y with z and then combining x with the result. The two expressions contain the same variables and operation symbol in the same left-to-right order. Yet their construction histories differ.

A machine that stores only the visible sequence

x · y · z has lost the distinction between the two possible compositions. Parentheses preserve that distinction. The oracle therefore learns that syntax is not merely a row of marks. It has shape. A term can be represented as a tree. The leaves are variables or constants. The internal nodes are operations. For the term

(x · y) · z the root applies · to two inputs. Its left input is itself the result of applying · to x and y. Its right input is z.

For the term

x · (y · z) the root again applies ·, but now the right branch contains the nested operation. The visible expression is a linear encoding of a branching structure. This matters because algebraic laws often compare different trees. Associativity does not compare two random strings. It compares two ways of composing the same three inputs:

```
(x · y) · z = x · (y · z)
```

The equation says that a particular structural distinction between trees may be ignored whenever the operation satisfies the law. Before that law is known, the distinction must be preserved. This gives an important rule for the virgin oracle: Do not compress syntax before discovering which syntactic differences are semantically irrelevant. If the machine omits parentheses from the beginning, associativity has been silently assumed. The system would no longer be virgin. A mathematical law would have been inserted into the notation before the oracle had an opportunity to discover it. The same danger appears in many familiar conventions. Humans routinely write: xyz without parentheses because multiplication is usually assumed associative. We write x^n because repeated multiplication has acquired a stable recursive pattern. We write fgh as a composition only after adopting a convention about direction and grouping.

Mature notation hides discoveries. A virgin oracle must begin with explicit structure and earn its abbreviations. The grammar of terms may be written schematically as:

```
t ::= x | c | u(t) | b(t,t)
```

Here x is a variable, c is a constant, u is a unary operation, and b is a binary operation. The

```
symbol ::= means that a term may be formed in any of the listed ways.
```

Yet even this notation contains prior design choices. It assumes the oracle already distinguishes variables, constants, and operations of specified arity. A more primitive machine might first encounter executable transformations and discover arity through experimentation. A unary operation accepts one input:

```
u : A → A
```

A binary operation accepts two:

```
b : A × A → A
```

A ternary operation accepts three:

```
q : A × A × A → A
```

The number of required inputs is the operation's arity. Arity is one of the earliest structural invariants of an operation. Relabeling the elements of A does not change the

number of inputs the operation accepts. Nor does changing the names of the operation symbols. Once the oracle tracks arity, it can reject malformed expressions before evaluating them. If b is binary, then b(x) lacks an input, while b(x,y,z) supplies too many. Grammar becomes a form of local verification. It prevents the machine from confusing expressions that merely contain familiar marks with expressions that can actually be executed. This distinction between syntax and execution is fundamental. A term is a formal construction.

Its evaluation requires a model. Consider the term:

```
t = (x · y) · z
```

To evaluate it, the oracle needs a carrier A, a specific binary operation on A, and an assignment of values to x, y, and z. Let:

```
v : {x,y,z} → A
```

be a variable assignment.

The value of a variable is:

```
⟦x⟧ᵥ = v(x)
```

The value of a composite term is determined recursively:

```
⟦s · t⟧ᵥ = ⟦s⟧ᵥ · ⟦t⟧ᵥ
```

The double brackets mean "the value of this expression under the chosen interpretation and assignment." The machine now possesses two distinct worlds. The syntactic world contains terms: x y x · y (x · y) · z x · (y · z) The semantic world contains their evaluated outputs in a particular structure. A term may be the same syntactic object while receiving different values in different structures. Conversely, two different terms may receive the same value under every assignment in one structure. This separation allows the oracle to ask a genuinely algebraic question: When do two different formal constructions always evaluate identically? That question leads directly to equations.

Before reaching equations, however, the machine must understand substitution. Suppose the term t contains the variable x. Another term s may be substituted for x. Write:

```
t[x := s]
```

For example:

```
(x · y)[x := z · z] = (z · z) · y
```

Substitution replaces a variable uniformly while preserving the surrounding structure.

It must be defined recursively. A careless textual replacement could capture variables, disturb grouping, or produce malformed expressions. The machine's internal tree representation avoids these problems. Substitution is not merely a technical convenience. It is one of the engines of generality.

The expression

```
(x · y) · z = x · (y · z)
```

does not describe only three particular elements named x, y, and z. The variables may be replaced by arbitrary terms. If the law is valid, then substitution yields new valid instances:

```
((a · b) · c) · d = (a · b) · (c · d)
```

or

```
(u · v) · (w · q) = u · (v · (w · q))
```

The original law propagates through all larger contexts. A small equation can therefore govern an unlimited family of expressions. This propagation depends on congruence. If s and t are recognized as equivalent, then replacing s by t inside a larger term should preserve equivalence. If: s ≡ t then: u · s ≡ u · t and: s · u ≡ t · u More generally, equivalent terms remain equivalent when inserted into the same syntactic context. This compatibility converts isolated equations into a theory. The oracle's term world now possesses a free structure. Terms can be generated without imposing equations beyond those necessary for syntax. Every different construction tree remains distinct.

Let TΣ(X) denote the set of all terms generated from variables X using the operations specified by Σ. The expression TΣ(X) is called a term algebra. It is free because no equations have yet identified distinct terms. If the signature contains one binary operation, then: (x · y) · z ≠ x · (y · z) as formal terms. They may later become equal in a quotient theory, but they are not initially the same tree. Freedom means that an arbitrary assignment of the generators into any compatible algebra extends uniquely to an evaluation map. Given:

```
v : X → A
```

there is a unique homomorphism:

```
v̄ : TΣ(X) → A
```

that agrees with v on variables. The extension satisfies:

```
v̄(x) = v(x)
```

and:

```
v̄(s · t) = v̄(s) · v̄(t)
```

This is one of the first universal properties in the book. The free term algebra is characterized by the fact that every assignment of its generators extends uniquely to a structure-preserving map. The machine need not know the phrase universal property. It may discover that the same mapping pattern recurs whenever a structure is freely generated. The pattern is:

```
generators → arbitrary target values
```

followed by: unique structure-preserving extension

This is more powerful than a direct description of all terms. It identifies the free algebra through its relation to every possible target algebra. The discovery marks an early shift from internal construction to external behavior. A structure can be known by what maps out of it. Later, category theory will elevate this principle into a general method. At the present stage, it appears naturally from evaluation. The oracle can now distinguish three layers: the alphabet, the grammar, and the interpretation. The alphabet

determines which marks are available. The grammar determines which arrangements count as terms. The interpretation determines what those terms do in a chosen world.

In compressed form:

```
alphabet → strings → terms → values
```

Confusing these levels produces errors. A symbol is not automatically an operation. A grammatical expression is not automatically true. A true equation in one model is not automatically a universal law. A notation that humans find familiar is not automatically a primitive concept. The virgin oracle must keep these distinctions explicit. This also clarifies the role of the Unicode alphabet developed for the present book. The alphabet includes ordinary letters, numerals, relation symbols, arrows, operation signs, quantifiers, logical connectives, Greek letters, subscripts, and superscripts. It is sufficient to write a substantial portion of abstract algebra compactly.

But the inventory itself does not contain algebra. The symbol ≁ does not create isomorphism. The character ∈ does not create membership. The character ⊗ does not create tensor products.

The character Ω does not create a subobject classifier.

These marks acquire meanings only through their places in grammars, equations,

interpretations, and networks of use. A mark becomes mathematical when it participates in a stable operational role. The oracle may initially use an anonymous token for a binary operation. Later, after discovering multiple operations, it may assign different symbols according to their law profiles. One associative and commutative operation might receive +. Another associative operation might receive ·. A reversible unary operation might receive $^{-1}$. This naming is retrospective. The structure is discovered before the familiar notation is assigned. Such delayed naming protects the experiment from conceptual leakage. If the machine is given a symbol called "inverse" and an executable routine already satisfying inverse laws, it has not discovered inverses. It has merely used them.

A genuine discovery process would proceed differently. The machine observes that, for certain operations and certain distinguished elements e, each x is associated with some y satisfying:

```
xy = e
```

and perhaps also:

```
yx = e
```

It then notices that the assignment x ↦ y behaves systematically. Only afterward does it introduce a unary symbol such as x^{-1} to compress the recurring construction. The symbol appears because a pattern has become stable. The same applies to constants. An element e becomes worthy of a dedicated symbol only after the oracle observes:

```
ex = x
xe = x
```

for all x. Before that discovery, e is merely one element among others. Once the law is recognized, e acquires a role that is independent of its original label. Under any isomorphism f:

```
f(e) = e′
```

where e′ is the identity in the target structure. The identity is no longer identified by appearance. It is identified by behavior. This is a crucial transition from labels to definable roles. A raw element is named externally. A structural element is characterized internally. The machine can test whether a candidate property uniquely determines an element. If two elements e and e′ both satisfy the identity laws, then:

```
e = ee′ = e′
```

Thus an identity, when it exists, is unique. The oracle may discover this result experimentally and later prove it symbolically. The proof requires only the defining equations. It does not inspect the operation table. This is a new kind of knowledge. Enumeration says that every tested associative table has at most one two-sided identity. Proof says that the defining laws themselves force uniqueness in every model. The transition from empirical pattern to syntactic derivation is one of the central developments of mathematical intelligence. To support it, the oracle needs inference rules. At the minimum, it may use: reflexivity, symmetry, transitivity, substitution, and compatibility with operations.

From:

```
s = t
```

it may infer:

```
t = s
```

From:

```
r = s
```

and:

```
s = t
```

it may infer:

```
r = t
```

From:

```
s = t
```

it may replace s by t inside a larger expression. These rules define equational reasoning. The machine's proof system can remain small because substitution and congruence give equations extraordinary reach. Suppose the oracle accepts:

```
ex = x
xe = x
```

It can prove identity uniqueness:

```
e = ee′
```

because e′ is a right identity, and:

```
ee′ = e′
```

because e is a left identity. Therefore:

```
e = e′
```

The proof is a path of equalities. Each step has a reason. The machine can preserve those reasons as provenance. This suggests that a proof is not merely a final equality. It is a structured transformation from one term to another. A proof may be represented as:

```
s = t₁ = t₂ = ⋯ = t
```

Each link records an axiom, substitution, or previously proved result. Later, proofs themselves may become objects of comparison. Two proofs can reach the same conclusion through different paths. In ordinary equational logic, those differences are often ignored. In higher settings, they may become mathematically significant. Once again, the way the oracle stores sameness determines what it can later discover. If all proofs of s = t are collapsed into the bare fact that s equals t, proof identity disappears. If proofs are retained as paths, the system may later examine paths between paths. The seed of higher identity is present even in elementary rewriting.

The oracle's language must therefore balance simplicity with future extensibility. A raw text representation may suffice for printing, but internally the system should preserve: the term tree, the operation arities, the variable bindings, the derivation history, the model in which evaluation occurred, and the transformation witnessing each equivalence. The printed equation is only the visible surface of a richer object. This distinction between surface notation and internal structure mirrors human mathematics. A short equation may summarize a long chain of definitions, conventions, and implicit type information. Experts read more than the visible marks. They reconstruct the surrounding architecture.

The virgin oracle must build that architecture explicitly. Typing provides one way to prevent meaningless combinations. Suppose x belongs to A and f maps A to B: x : A

```
f : A → B
```

Then f(x) belongs to B:

f(x) : B If g expects an input from C, the expression g(f(x)) is invalid unless B and C are appropriately connected. Types make compatibility visible. At an early finite stage, the oracle may work in a single carrier, so every operation accepts and returns elements of the same set. This is the single-sorted setting of ordinary universal algebra. Later, multiple sorts become necessary. Scalars and vectors behave differently. Objects and arrows occupy different roles. Regions, local sections, and restriction maps have different types. Types prevent the machine from forming expressions that are syntactically plausible but structurally incoherent.

A multisorted signature may include: r : R x : M addition on R, addition on M, multiplication on R, and scalar action from R × M to M. The expression rx is meaningful, but xr may not be defined. The oracle discovers that not all multiplication-like operations are interchangeable. Their input and output sorts matter. Typed grammar therefore becomes a map of permitted composition. This prepares the transition to categories, where arrows compose only when codomain and domain match: $A \longrightarrow^{f} B \longrightarrow^{g} C$ Then:

```
g ∘ f : A → C
```

but f ∘ g may be undefined.

Associativity survives, but total binary composition is replaced by partial composition governed by types. The path from untyped terms to typed arrows is not a departure from algebraic generation. It is a refinement of grammar in response to structural constraints. The same principle governs the entire development: A language should grow only when the current language cannot express a recurring invariant economically and correctly. This protects the virgin oracle from being handed a finished ontology. It must earn each new category of symbol through use. The emergence of mathematics from marks can now be summarized more precisely: Σ generates strings.

Grammar selects terms. Types restrict admissible composition. Models assign operations and values. Evaluation connects terms to outcomes. Substitution produces generality. Equations compare terms. Congruence propagates equality through contexts. Proofs record justified transformations. Quotients convert proved equivalences into new objects. Universal properties characterize the free structures from which the process began. The development forms a loop. The free term algebra generates all formal possibilities. Equations then identify some possibilities. The quotient becomes a new algebra. That algebra can itself supply generators for another term system.

Thus the movement from marks to terms is not merely an introductory stage. It is repeated at every level of mathematical growth. Categories generate diagrams. Diagrams generate cones.

Local regions generate covers. Covers generate descent data. Types generate terms. Terms generate identity types. Identity types generate higher paths. At every stage, the oracle constructs a grammar for a newly discovered kind of object. The finite alphabet remains small. The space of recursively generated structure becomes vast. This is the power of compositionality. A finite set of primitive marks and formation rules can generate unbounded formal complexity. But unbounded generation alone produces chaos. The next step is to discover why a small number of equations can organize that chaos into coherent worlds. That is the subject of the next chapter.

Chapter 5: Why Equations Compress Worlds

Adapted from The Virgin Oracle, Chapter 3 (2026).

A finite alphabet can generate indefinitely many expressions. A grammar can ensure that those expressions are well formed. Evaluation can assign them values in a chosen structure. Yet none of these capacities explains why mathematics becomes organized rather than merely enormous. The organizing force is the equation. An equation identifies two expressions that were constructed differently but produce the same result under specified conditions. It turns recurrence into a reusable law. Instead of storing every successful calculation separately, an intelligence stores a compact relation that governs an entire family of calculations. The elementary form is:

```
s = t
```

Here s and t are terms. The equation asserts that they should be treated as interchangeable within the relevant theory. This assertion can have several meanings. It may mean that s and t evaluate to the same result in one particular structure. It may mean that they agree in every structure within a chosen class. It may mean that t can be derived from s using accepted axioms. It may even be adopted as a defining relation that creates a new quotient of the term world. The visible equation is the same in each case. Its epistemic status is not. The virgin oracle must therefore record not only an equation but the reason the equation is present. An equation may be: observed in one experiment, verified exhaustively in a finite domain, supported statistically across sampled structures, assumed as an axiom, derived from previous equations, or validated in every model by a proof.

Without such provenance, the oracle would confuse local coincidence with universal necessity.

Consider a binary operation on a three-element carrier. For one particular table, the equation

```
xy = yx
```

may hold for every x and y. The oracle may then record that the operation is commutative. But it must not conclude that every binary operation is commutative. The complete three-element atlas immediately supplies counterexamples. The equation divides the atlas into two regions:

```
Mod(xy = yx)
```

and its complement. Thus an equation is not merely a statement. It is a classifier of worlds. Given an equation E, define:

```
Mod(E) = {A | A satisfies E}
```

If E contains several equations, then a model must satisfy all of them:

```
Mod({E₁, E₂, …, Eₙ}) = Mod(E₁) ∩ Mod(E₂) ∩ ⋯ ∩ Mod(Eₙ)
```

Adding equations reduces the class of permitted models:

```
E ⊆ F ⇒ Mod(F) ⊆ Mod(E)
```

A stronger theory admits fewer worlds. This reversal is fundamental. More syntax produces fewer models. A theory gains precision by excluding possibilities. Suppose the oracle begins with no equations governing a binary operation. Every table is permitted. It then accepts associativity:

```
(xy)z = x(yz)
```

All nonassociative tables are excluded. If it next adds commutativity:

```
xy = yx
```

the permitted family shrinks again. If it adds idempotence:

```
xx = x
```

it isolates a still smaller family. The process resembles carving structure from possibility.

The initial term algebra contains every formal expression. The initial model universe contains every operation table. Equations remove distinctions in syntax while excluding incompatible worlds in semantics. These two effects move in opposite directions: On the syntactic side, equations identify more terms. On the semantic side, equations admit fewer models. This dual movement is one of the central architectures of formal mathematics. Let T(X) be the free term algebra. A set of equations E generates an equivalence relation ≡E on terms. The quotient T(X)/≡E contains fewer distinct formal objects than T(X), because terms proved equivalent are collapsed into one class.

At the same time, Mod(E) contains fewer structures than the universe of all structures of the given signature. Equations therefore compress syntax and constrain semantics. The oracle can exploit both effects.

Compression by substitution

A single equation represents far more than one comparison. Take associativity:

```
(xy)z = x(yz)
```

Because x, y, and z are variables, they may be replaced by arbitrary terms. Substitution yields infinitely many instances. For example:

```
((ab)c)d = (ab)(cd)
```

This follows by substituting ab for x, c for y, and d for z in the original law. Another instance is:

```
a((bc)d) = a(b(cd))
```

The machine does not need to store each instance independently. It stores the general law and a substitution rule. The compression ratio becomes enormous. A finite equation governs an unbounded family of term trees. The more contexts in which the equation can be applied, the more expressive power it possesses. This helps explain why equations containing variables are so central to algebra. They do not describe isolated objects. They define patterns invariant under substitution. A table of observations might contain millions of equal evaluations. A single universally quantified identity can replace them:

```
∀x,y,z, (xy)z = x(yz)
```

The quantifier need not always be written. In algebra, variables appearing in identities are conventionally understood to range over all elements of the structure. But the virgin oracle should retain the quantification explicitly in its internal representation. Otherwise, it might confuse an identity holding for every assignment with an equation holding only for one selected triple. The difference is substantial: a particular fact:

```
(ab)c = a(bc)
```

a universal law:

```
∀x,y,z, (xy)z = x(yz)
```

The first concerns three chosen elements. The second governs the operation itself. The transition from particulars to variables is an act of abstraction. Variables erase the identities of particular inputs while preserving their structural positions. The equation says that the result does not depend on which elements occupy those positions.

The economy of laws

Suppose the oracle observes a collection D of operation tables and their evaluations. It seeks a theory E that describes them economically. One possible objective is:

```
cost(E,D) = length(E) + unexplained(D | E)
```

The first term penalizes a complicated theory. The second penalizes observations not accounted for by the theory. A theory containing no laws has almost zero descriptive cost but explains little. A theory that lists every observed table exactly explains everything but is nearly as long as the data itself. A useful theory lies between these extremes. It achieves compression by exploiting recurrence. The machine may compare two candidate descriptions. Description 1 stores every value of every expression encountered. Description 2 stores the operation table, a few equations, and rules for deriving the rest. If Description 2 is substantially shorter while reproducing the same verified outcomes, the equations have explanatory value.

This resembles the minimum-description-length principle: prefer a compact model that accounts for the data without unnecessary complexity. Yet mathematics cannot be reduced to compression alone. A false theory may compress a limited dataset more efficiently than a true one. If the observations omit the cases where the law fails, the machine may adopt an elegant but incorrect conjecture. For this reason, compression must be paired with adversarial testing. The oracle should actively seek cases that distinguish competing theories. It should not merely collect confirming examples. It should construct counterexamples. The discovery cycle becomes: propose a short law, identify all predicted consequences, search the unexplored space for violations, retain the law only within the domain where it survives.

The better the oracle becomes, the more aggressively it attempts to falsify its own compressions. A conjecture is valuable partly because it tells the machine what kind of counterexample to search for.

Counterexamples as structural information

A counterexample is not merely a failed theorem. It maps the boundary of a concept. Suppose the oracle conjectures:

```
xy = yx ⇒ (xy)z = x(yz)
```

It is claiming that every commutative operation is associative. The complete finite atlas can test the implication. If a table satisfies commutativity but fails associativity, the oracle obtains an explicit countermodel. That table proves that the premise is insufficient. The machine should preserve the counterexample in a minimal form: carrier size, operation table, assignment of variables witnessing failure, laws satisfied, law violated. For example, it may report:

```
xy = yx holds for all x,y,
but for x = a, y = b, z = c,
```

(xy)z ≠ x(yz). This is more informative than a simple label reading FALSE. The counterexample reveals how the proposed implication fails. The oracle can then ask whether an additional premise repairs the claim. Perhaps commutativity together with another law implies associativity in the tested domain. Perhaps no short addition works. Perhaps the smallest counterexample requires four elements. Each result enriches the theory landscape. Counterexamples therefore perform several roles: They prevent overgeneralization. They demonstrate independence between laws.

They locate minimum model sizes for failure. They suggest missing hypotheses. They reveal alternative structures excluded by the proposed theory. A mature mathematical library should store counterexamples alongside theorems. They define the shape of the possible.

Equations as partitions of model space

Every equation divides a model universe into those structures that satisfy it and those that do not. A collection of equations assigns each model a law signature. For candidate equations E_1, E_2, …, E_m, define:

```
v(A) = (b₁, b₂, …, bₘ)
```

where:

```
bᵢ = 1 if A satisfies Eᵢ,
bᵢ = 0 otherwise.
```

Two operation tables with the same signature satisfy exactly the same tested laws. They may still be nonisomorphic, but the current vocabulary does not distinguish them. The law signature is therefore an observational representation relative to the chosen equation set. As the oracle generates more equations, signatures become more discriminating. Some previously indistinguishable structures separate. Others remain identical under every tested law. This leads to a hierarchy of observational resolution. At depth one, the machine may test only very short equations such as:

```
xx = x
xy = yx
```

At greater depth, it tests longer terms:

```
(xy)z = x(yz)
x(yx) = (xy)x
(xy)(zx) = x((yz)x)
```

Each increase in term depth expands the observational vocabulary. Two structures may agree on all equations up to depth three but differ at depth four. Thus finite law mining produces approximations to complete equational theory. The oracle must avoid claiming that agreement within a bounded language means total equivalence. It should say: indistinguishable by equations of the tested signature, variable count, and depth. This careful wording preserves the difference between computational evidence and mathematical completion.

Theory closure

Once equations have been accepted, other equations may follow from them. Let C(E) denote every equation derivable from E using the oracle's inference rules. Then: $E \subseteq C(E)$ If $E \subseteq F$, then: $C(E) \subseteq C(F)$ And:

```
C(C(E)) = C(E)
```

These are the laws of a closure operator. The first says that the closure contains the original equations. The second says that stronger assumptions cannot produce fewer consequences. The third says that deriving all consequences twice yields nothing beyond deriving them once. A theory is deductively closed when:

```
C(E) = E
```

In practice, no finite machine stores every member of an infinite closure explicitly. Instead, it stores a finite basis, derivation procedures, normal forms, or proof-search mechanisms. This creates a second compression.

The initial equation compresses observations. A finite axiom basis compresses an entire deductively closed theory. For example, a small collection of group laws generates indefinitely many consequences:

```
e⁻¹ = e
(x⁻¹)⁻¹ = x
(xy)⁻¹ = y⁻¹x⁻¹
xy = xz ⇒ y = z
```

The oracle should not store these as unrelated facts. It should record their dependency on the underlying axioms. A theorem is then a compressed path through the closure.

Redundancy and minimal axioms

An equation may be true in every model of a theory but unnecessary as an axiom because it follows from the others. Suppose:

```
E = {E₁, E₂, E₃}
```

If:

```
C({E₁,E₂}) = C(E)
```

then E_3 is redundant. The oracle should search for smaller bases generating the same theory. A minimal axiom set satisfies two conditions: It generates the target closure. Removing any one axiom changes that closure. Minimality may be measured in several ways. One basis may use fewer equations but longer terms. Another may use more

equations that are individually simpler. A third may be easier for automated proof search. Thus there may be no single best basis. The machine can evaluate:

number of axioms, total symbol length, maximum term depth, proof-search cost, number of models excluded per axiom, and transportability to related theories. This shows that mathematical elegance has multiple dimensions. A short basis is not automatically computationally convenient. A redundant axiom may drastically shorten proofs. Human textbooks often include derived laws because they clarify structure, even when they are logically unnecessary. The virgin oracle should distinguish: logical necessity, descriptive convenience, computational utility, and conceptual centrality. These are related but not identical.

Rewriting and normal forms

An equation can be used symmetrically:

```
s = t
```

But calculation often requires a direction:

```
s → t
```

This turns an equation into a rewrite rule. For example:

```
ex → x
x⁻¹x → e
(xy)⁻¹ → y⁻¹x⁻¹
```

A rewrite system repeatedly transforms terms until no rule applies.

Write: s ↠ t when s reaches t through zero or more rewrite steps. If every term eventually stops rewriting, the system is terminating. If every pair of rewrite paths from the same term can be reconciled, the system is confluent:

```
s ↠ u and s ↠ v ⇒ ∃w, u ↠ w and v ↠ w
```

When termination and confluence both hold, each term has a unique normal form n(s). Then equality becomes computational:

```
s ≡E t ⇔ n(s) = n(t)
```

Instead of searching arbitrary proofs, the oracle reduces both expressions and compares their normal forms. This is a powerful form of compression. An entire equivalence class is represented by one canonical term. Yet orienting equations is not always straightforward. Associativity may be directed as:

```
(xy)z → x(yz)
```

This always moves parentheses to the right and therefore terminates for finite terms. But commutativity:

```
xy = yx
```

cannot be oriented naively in both directions without causing endless reversal:

```
xy → yx → xy → yx → ⋯
```

The machine needs an ordering on terms. It may rewrite the larger term toward the smaller according to a fixed canonical order. For example, if x is ordered before y, it may direct:

```
yx → xy
```

This selects one representative from each commutative pair.

When associativity and commutativity interact, normal forms become more subtle. Parentheses disappear structurally, and factors may need to be sorted. More complex theories require completion procedures that add new rewrite rules to resolve conflicts. The oracle can learn from these conflicts. A failed normalization attempt indicates hidden interaction among equations. A critical pair occurs when two rewrite rules apply to overlapping parts of the same term and lead in different directions. Resolving all critical pairs is one route toward confluence. Thus theorem discovery may emerge from the effort to make calculation deterministic. New equations are introduced because existing rewrite rules disagree.

The desire for unique normal forms generates mathematical consequences.

The distinction between identity and definition

Not every equation states a discovered law. Some equations introduce notation or define a new object. For example:

```
x² = xx
```

This may be a definition of exponent notation rather than a theorem requiring proof. Similarly:

```
[x,y] = xyx⁻¹y⁻¹
```

defines the commutator. And:

```
ker f = {x | f(x) = e}
```

defines the kernel. The virgin oracle must distinguish definitional equality from proved equality. A definitional equation expands a new symbol into an existing construction. It is true by the adopted meaning of the symbol. A theorem asserts that two independently meaningful expressions coincide as a consequence of structure.

```
The distinction can be represented internally even if both are printed using =.
```

Without it, the machine might "prove" statements merely by unfolding hidden definitions and mistake abbreviation for discovery. A mature system may use different internal relations: ≔ for definition,

```
= for equality within a structure,
```

equivalence signs can distinguish syntax; ~= denotes isomorphism; ~ can denote broader equivalence. The exact symbols matter less than the separation of roles. Different grades of sameness should not be collapsed prematurely.

Equations can create worlds

So far, equations have described existing operation systems. But equations can also be used constructively. Start with a free term algebra T(X). Impose equations E. Form the quotient:

```
A = T(X)/≡E
```

The resulting algebra is presented by generators X and relations E. Write:

```
A = ⟨X | E⟩
```

The generators supply raw possibilities. The relations identify expressions. For example, one might begin with a generator a and impose:

```
aⁿ = e
```

The quotient creates a cyclic structure whose repeated powers close after n steps, subject to the precise surrounding axioms. The important point is that the structure need not first exist as a complete table. It can be created formally from a presentation. This changes the oracle's role. It no longer merely classifies given worlds. It designs worlds by selecting generators and relations.

Every presentation is an experiment in controlled identification. Too few relations leave a large free structure. Too many relations may collapse the structure severely, perhaps identifying every element. The oracle can explore the effect of each relation on the quotient. It can ask: Which expressions become equal? How many elements remain in finite cases? Which symmetries survive? Which maps into other structures are permitted? Which relations are redundant? The space of presentations becomes a laboratory of theory creation.

Equations and transformations

An equation may also express the invariance of an operation under transformation. If f preserves a binary operation, then:

```
f(xy) = f(x)f(y)
```

This equation defines a homomorphism. Unlike associativity, which relates two expressions inside one structure, the homomorphism law relates evaluations across two structures. The map f transports products from the source to the target without changing their operational meaning. This creates a new criterion of sameness. Two structures are isomorphic when there is a reversible homomorphism between them. Thus equations do not only constrain elements. They constrain maps. Later, naturality equations will constrain families of maps:

```
G(f) ∘ ηA = ηB ∘ F(f)
```

Cocycle equations will constrain gluing transformations:

```
φᵢₖ = φⱼₖ ∘ φᵢⱼ
```

Coherence equations will constrain transformations among transformations.

The same basic device-equating two composites-organizes increasingly high levels of structure. Abstract algebra expands by changing what the terms denote while retaining the architecture of equation. At first, terms denote elements. Later, they denote functions. Later still, they denote functors, transformations, paths, and higher cells. The equation persists as a declaration that two routes through a construction agree.

Commutative diagrams as equations

A diagram is commutative when every permitted path with the same starting and ending objects yields the same composite. Suppose: A $\longrightarrow^{f}$ B A $\longrightarrow^{h}$ C B $\longrightarrow^{g}$ C The triangle commutes when:

```
g ∘ f = h
```

The diagram is a visual equation. More complicated diagrams compress several equations at once. Their geometry helps humans perceive compatibility among maps. For a machine, the diagram may be represented as a graph together with path-equality constraints. The oracle can then ask whether a proposed diagram commutes in every model, in one model, or only under additional assumptions. This extends equation mining from term trees to path expressions. The same compression principle applies. Rather than storing all route comparisons separately, the machine records a structural diagram whose commutativity governs them. The rise of category theory can therefore be seen as a migration of algebraic equation from operations on elements to compositions of arrows.

Local equations and gluing

Equations also decide whether local pieces fit together. Suppose local data s_i and s_j are defined on overlapping regions U_i and U_j. Compatibility requires:

$$s_i \text{ restricted to } U_i \cap U_j = s_j \text{ restricted to } U_i \cap U_j$$

This is an equation on an overlap. When every compatible family has a unique global realization, local equations compress a global object. The global information is not listed independently. It is reconstructed from pieces and compatibility conditions. Thus gluing is another form of equation-driven compression:

$$\text{local data} / \text{overlap agreement} = \text{global structure}$$

The slash again represents disciplined identification rather than numerical division. At higher levels, pairwise agreement is insufficient. Transition maps must satisfy a cocycle condition on triple overlaps:

$$\varphi_{ik} = \varphi_{jk} \circ \varphi_{ij}$$

This equation says that two routes of identification agree. The pattern is the same as associativity and diagram commutativity. Mathematical coherence repeatedly appears as equality between alternative compositions.

Why some equations become foundational

The oracle may discover millions of valid equations. Why should a few become the backbone of its library? A foundational equation tends to possess several properties at once. It is short. It holds in a large and varied family of models. It interacts productively with other laws. It supports recursive construction. It survives transport through important maps.

It allows canonical forms or efficient calculation. It appears again at higher structural levels. Associativity is exemplary because it satisfies all of these. The same equation governs: products of elements, composition of functions, composition of homomorphisms, composition of arrows, concatenation of paths up to suitable equality, and higher compositions under coherence. Its reappearance suggests that the equation is not tied to one kind of object. It expresses a general architecture of sequential combination. Distributivity is similarly powerful because it coordinates two operations:

$$x(y + z) = xy + xz$$

It permits one structure to act coherently upon another. It supports expansion, linearity, modules, representations, and algebra over scalars. The homomorphism equation:

```
f(xy) = f(x)f(y)
```

is foundational because it defines preservation itself. The sheaf equation is foundational because it converts compatible local descriptions into a global object. Univalence is foundational because it reorganizes the relation between equivalence and identity. The importance of such equations lies not only in how much data they compress, but in how many conceptual levels they connect. The oracle should therefore track recurrence across domains. When the same formal pattern appears in several previously separate theories, it may deserve promotion to a higher abstraction.

This is how category theory could emerge. The oracle notices that products of sets, groups, spaces, and other structures are all characterized by analogous mapping equations. It extracts the common form. Abstraction then becomes second-order compression: first compress observations into equations, then compress families of equations into universal patterns.

The danger of overcompression

Compression can fail by erasing significant distinctions. Suppose two proofs establish the same equation. At the level of ordinary theorem truth, they may be interchangeable. But one proof may be constructive and the other nonconstructive. One may reveal an algorithm. One may generalize to a broader setting. One may preserve geometric information lost by the other. If the oracle stores only the endpoint equation, it may discard the most valuable part of the discovery. Similarly, collapsing isomorphic objects into one class may erase their automorphisms. Collapsing equivalent categories into one point may erase the functors and natural transformations witnessing equivalence. Collapsing homotopic paths may erase higher homotopies.

The oracle must therefore decide what level of compression is appropriate. A quotient set retains classes but forgets internal symmetry. A groupoid retains objects and reversible maps. A higher groupoid retains maps among maps. Compression should not be maximal. It should preserve the structure required for future questions. This gives a more careful principle: A good abstraction removes distinctions that are irrelevant to the current theory while retaining the witnesses needed for the next theory. The word current matters. A distinction irrelevant today may become important after the language expands.

The oracle should therefore preserve a recoverable lower-level representation whenever possible. It may work with canonical representatives for efficiency while retaining the orbit and its symmetry data in storage. Mathematical intelligence requires reversible abstraction where feasible.

From equations to theories

An isolated equation is rarely a complete theory. Structures arise from packages of interacting laws. A theory E may contain:

```
E₁: (xy)z = x(yz)
E₂: ex = x
E₃: xe = x
E₄: x⁻¹x = e
```

```
E₅: xx⁻¹ = e
```

The machine can study the model class selected by this package. It can determine which equations follow, which axioms are redundant, which finite models exist, and which maps preserve the resulting structure. It can also compare theories. Say E is weaker than F when every model of F is a model of E: Mod(F) ⊆ Mod(E) Equivalently, F entails every equation in E. Theories therefore form an ordered landscape. Moving upward by adding laws produces more specialized theories. Moving downward by removing laws produces more general theories. Meet-like operations combine requirements. Join-like operations combine deductive consequences. The oracle can construct a lattice or partial order of theories, at least within a bounded equation universe.

This landscape reveals relations hidden by traditional naming. Two named structures may differ by one axiom. One theory may possess several minimal bases. A law thought central may turn out to follow from a different package in finite models. The machine's anonymous theory graph may expose the architecture of algebra without relying on historical disciplinary boundaries. Groups, semigroups, quasigroups, lattices, loops, rings, and modules are not isolated dictionary entries. They occupy connected regions in a space of laws and constructions. The library becomes navigable through implication.

The equation as an instrument of intelligence

An equation performs several functions simultaneously. It records recurrence. It enables substitution. It generates a congruence. It constrains models. It supports proof. It defines quotients. It characterizes maps. It imposes coherence. It permits compression. It creates new objects. This multiplicity explains why equation is one of the central inventions of formal thought. Yet the equation is not the endpoint. It is a hinge between generation and identification. On one side lie many formal possibilities. On the other side lie structured classes, canonical forms, and invariant objects. The equation decides which differences can be crossed. The virgin oracle begins with expressions that are distinct because they were built differently.

Through evaluation, testing, proof, and abstraction, it discovers that some of those differences do not affect behavior.

It then records:

```
s = t
```

That small mark of equality may collapse an infinite family of distinctions. The resulting compression is not merely economical storage. It creates a new world in which s and t are no longer separate objects. Mathematics advances whenever such a collapse reveals more structure than it destroys. The oracle must learn to perform that act carefully, preserving assumptions, counterexamples, and witnesses. When it succeeds, an equation becomes more than a statement about a world. It becomes an operation that reorganizes the world. The next chapter enters the first complete experimental cosmos in which this process can be carried out exhaustively: the universe of finite binary operations.

The Emergence of Algebra

Chapter 6: The Complete World of Finite Operations

Adapted from The Virgin Oracle, Chapter 4 (2026).

The virgin oracle now has an alphabet, a grammar, a term language, and a method for testing equations. It can distinguish syntax from interpretation and observation from proof. What it still lacks is a sufficiently rich world in which laws can be discovered without being supplied in advance. The most useful first laboratory is not the natural numbers, the integers, or familiar arithmetic. Each of those already carries a large amount of inherited structure. Addition on the integers is associative, commutative, and equipped with an identity and inverses. Multiplication interacts with addition through distributivity. Beginning there would expose the oracle to successful structures but conceal the space of alternatives against which those structures become significant.

A genuine discovery environment should include failures as well as successes. The oracle should see operations that are associative and operations that are not. It should encounter identities, one-sided identities, multiple idempotents, absorbing elements, cancellation, noncancellation, symmetry, asymmetry, closure, and pathological combinations. It should learn each property by comparison with nearby structures where that property fails. A complete finite atlas provides such a world. Let A be a carrier with n distinct elements:

```
A = {0, 1, …, n-1}
```

A binary operation on A is a function:

```
· : A × A → A
```

For every ordered pair (x,y), the operation chooses one output x·y in A. Since there are n^2 ordered pairs and n possible outputs for each pair, the total number of labeled binary operations is: $n^{(n^2)}$

```
For n = 1, there is only:
1¹ = 1
```

binary operation.

```
For n = 2, there are:
2⁴ = 16
```

binary operations.

```
For n = 3, there are:
3⁹ = 19,683
```

binary operations.

```
For n = 4, the count becomes:
4¹⁶ = 4,294,967,296
```

The jump is severe. Exhaustive enumeration remains trivial at size two, manageable at size three, and much more demanding at size four. This sharp growth is not a defect. It reveals why a complete three-element universe is an unusually valuable seed. It is small

enough to search in full yet large enough to exhibit substantial algebraic diversity. The oracle is given no names for the operations. Each table receives only an anonymous identifier. An operation may be stored as a table: · 0 1 2 0 0 1 2 1 1 2 0 2 2 0 1 The row identifies the first input, the column the second, and the entry the output. The same operation may be encoded as a sequence: 0,1,2,1,2,0,2,0,1 The sequence is not mathematically privileged. It is simply a convenient machine representation. The operation itself is the mapping from ordered pairs to outputs.

This distinction matters because different encodings may describe the same abstract structure. A row-major sequence, a column-major sequence, a graph, and a collection of equations may all present one operation differently.

The oracle's initial task is not to choose the most elegant representation. It is to preserve enough information for exact evaluation. Given a table, every term receives a value under every assignment of variables. For example, to evaluate: (xy)z the oracle first computes x·y, then multiplies the result by z. To evaluate: x(yz) it first computes y·z, then combines x with that result. Associativity is present precisely when the two evaluations agree for every triple:

```
∀x,y,z ∈ A, (xy)z = x(yz)
```

For a three-element carrier, only 27 assignments must be tested. Thus every one of the 19,683 operations can be checked exhaustively. Commutativity requires nine comparisons:

```
∀x,y ∈ A, xy = yx
```

Idempotence requires three:

```
∀x ∈ A, xx = x
```

A left identity is an element e satisfying:

```
∀x ∈ A, ex = x
```

A right identity satisfies:

```
∀x ∈ A, xe = x
```

A two-sided identity satisfies both. An absorbing element z satisfies:

```
∀x ∈ A, zx = xz = z
```

Left cancellation means:

```
∀x,y,z ∈ A, xy = xz ⇒ y = z
```

Right cancellation means:

```
∀x,y,z ∈ A, yx = zx ⇒ y = z
```

Each property can be stated without assigning a human name. The oracle may initially record only equations or implication patterns. For example:

```
E₁: (xy)z = x(yz)
E₂: xy = yx
E₃: xx = x
E₄(e): ex = x
E₅(e): xe = x
```

The symbol e in E_4 and E_5 differs from the universally quantified variables x,y,z. It represents a candidate element whose existence must be tested. Thus identity is not a pure identity in the same form as associativity unless the language already contains a distinguished constant e. There are two possible discovery strategies. The oracle may search for elements possessing a role:

```
∃e, ∀x, ex = xe = x
```

Or it may enlarge the language by adding a constant symbol e and study structures in which that constant is interpreted. The first strategy discovers structure inside an unpointed operation table. The second studies an expanded structure containing both an operation and a distinguished element. The distinction becomes important throughout algebra. A group may be described as a set with an operation, an identity element, and an inverse operation. Alternatively, one may begin with a binary operation and define identity and inverses by existential properties when they exist. The second approach is convenient for equational axiomatization. The first is more appropriate for a virgin oracle examining anonymous tables.

The oracle should therefore begin by discovering roles before naming constants.

The first census

The most elementary use of the atlas is counting. How many operations satisfy E_1?

How many satisfy E_2? How many satisfy E_1 and E_2? How many have a two-sided identity? How many contain an absorbing element? How many are idempotent? How many satisfy cancellation? These counts provide a first map of the operation universe. Yet raw counts are not enough. A property may appear frequent because many labeled copies of one structure are present. If an operation has several relabelings, each appears as a separate table in the complete labeled atlas. The oracle must eventually distinguish: the number of labeled tables, the number of isomorphism classes, and the distribution of automorphism groups. At the beginning, however, even labeled counts are informative. They show that laws occupy regions of different sizes.

Some laws are extremely common. Others are rare. A rare law is not automatically important. Nor is a common law automatically trivial. Frequency supplies one signal among several. Suppose a law holds in almost every table. It may express a weak constraint. Suppose another holds in only a tiny family. It may characterize a highly organized structure or merely an accidental curiosity. The oracle must compare frequency with consequence. A law is structurally valuable when it changes what can be constructed or inferred. Associativity, for example, has an effect extending far beyond its frequency in small tables. It permits iterated products to be treated coherently. A table satisfying associativity can support words, powers, subsemigroups, ideals in richer settings, and homomorphisms whose compositions remain within the same theory.

The law generates architecture.

The census therefore becomes a theory of consequences. For each operation A, the oracle stores a law profile:

```
L(A) = {E | A |= E}
```

For a bounded equation library $E_1,\dots,E_m$, this becomes a finite bit vector:

```
v(A) = (b₁,…,bₘ)
```

Two operations with the same vector are indistinguishable by the currently tested laws. They may nevertheless differ structurally. This reveals an important limitation of finite equation mining. A vocabulary determines what the oracle can see. If the vocabulary contains only one-variable equations, then operations differing only in multivariable behavior may appear identical. If it contains terms of depth at most two, then differences requiring deeper nesting remain invisible. The machine should therefore record not simply that two operations share a law profile, but that they share a profile relative to a stated search boundary. For example: same equations using at most three variables and term depth at most four This is an empirical equivalence, not complete algebraic equivalence.

Generating candidate laws

The space of possible equations grows rapidly. Even with one binary operation and three variables, the number of terms increases with depth. Depth zero terms are: x, y, z Depth one includes: xx, xy, xz, yx, yy, yz, zx, zy, zz Depth two includes every binary combination of shallower terms: (xx)x x(xy)

(xy)(zx) and many others. If every pair of terms becomes a candidate equation, the search quickly fills with uninformative statements. The oracle needs rules for prioritization. The first rule is type compatibility. Both sides of an equation must have the same output type. The second is variable compatibility. An equation such as:

```
x = y
```

is not universally true in nontrivial structures and will fail immediately, but it is still syntactically legitimate. Other equations may contain a variable on only one side:

```
xy = x
```

Such laws can be significant, as in left-zero operations. The oracle should not require the same variables to appear on both sides. That restriction would exclude important theories. A better filter is descriptive economy. Short terms are tested before long ones. The search might proceed by total symbol length: x xx xy (xy)z x(yz) and so forth. The oracle can also avoid equations that are merely reflexive syntactic identities:

```
s = s
```

It may remove obvious renamings, so that:

```
xy = yx
```

is not treated as different from:

```
yx = xy
```

Likewise, variable permutations can place equations into a canonical form. For instance:

```
xy = yx
```

and:

```
xz = zx
```

express the same schema after renaming variables. Canonicalization reduces redundancy before evaluation. Yet variable symmetry must be handled carefully. The equations:

```
xy = x
```

and:

```
xy = y
```

are transformed into one another by exchanging variable names and simultaneously exchanging input positions. If the operation itself is not assumed commutative, those laws describe different orientations: left projection and right projection. Canonicalization should preserve structural distinctions while eliminating mere notational duplication. This requires the oracle to understand the action of variable permutations on terms. Again, group action enters before group theory has been named.

Law discovery as partition refinement

Every tested equation divides the operation atlas. Suppose all operations initially lie in one block. Testing commutativity separates them into: commutative operations noncommutative operations Testing associativity divides each block again. Testing idempotence refines the partition further. After m equations, operations are grouped by their bit vectors. The oracle can measure how much each equation refines the current partition.

An equation that every operation either satisfies or violates uniformly adds no discrimination at that stage. An equation that splits a large block into balanced subfamilies may be highly informative. One possible information score is based on entropy reduction. But mathematical significance is not identical to balanced classification. A law may isolate a small, exceptionally generative family. Therefore the oracle should combine several measures: how strongly the law partitions models, how short the law is, how many other laws it helps predict, how stable it remains across carrier sizes, and what constructions it enables. The atlas becomes a testing ground for competing notions of significance.

A purely statistical intelligence may favor equations dividing the data efficiently. A mathematical intelligence should also recognize laws that create reusable form.

The emergence of identities

Among the earliest structural roles likely to be discovered is the identity element. For each operation table, the oracle tests each candidate e ∈ A. A left identity satisfies:

```
ex = x
```

for every x. In a table, this means the row labeled e reproduces the header sequence: 0,1,2 A right identity means the column labeled e reproduces the same sequence. A two-sided identity is both.

This is visually simple, but its importance is not merely that one row and one column have a recognizable pattern. The identity creates a neutral operation context. It permits an element to be inserted without changing the result. The oracle may notice that

identity elements, when they exist, are unique. Suppose e and f are both two-sided identities. Then:

```
e = ef = f
```

The first equality uses that f is a right identity. The second uses that e is a left identity. This proof does not depend on the number of elements or on exhaustive enumeration. It depends only on the defining role. The oracle has moved from atlas evidence to universal deduction. The discovery may occur in stages. First, the machine observes that no tested operation has two distinct two-sided identities. Next, it formulates the conjecture: An operation has at most one two-sided identity. Then it searches larger finite atlases or random samples and finds no counterexample. Finally, it constructs the symbolic proof. The proof reveals why no counterexample can exist.

This progression illustrates the intended relation between computation and reasoning. Enumeration suggests a theorem. Proof explains its necessity.

One-sided structure

The complete atlas prevents the oracle from assuming that left and right behavior coincide. An element may satisfy:

```
ex = x
```

without satisfying:

```
xe = x
```

Likewise, cancellation may hold on one side but not the other. An absorbing element may be left absorbing, right absorbing, or both.

Human learners often first encounter commutative operations, where left and right distinctions disappear. The atlas exposes the asymmetry directly. This is important because many advanced algebraic structures are noncommutative. Matrix multiplication, function composition, operator algebras, free groups, and categories all preserve order. A virgin oracle raised only on commutative examples might regard asymmetry as an exception. A complete operation universe reveals that commutativity itself is a special law. The machine learns not to import symmetry where none has been proved. This discipline later becomes essential in manipulating inverses:

```
(xy)⁻¹ = y⁻¹x⁻¹
```

The order reverses. A system that silently assumes commutativity would miss the distinction.

Idempotents and fixed behavior

An element x is idempotent when:

```
xx = x
```

An operation is globally idempotent when every element is idempotent:

```
∀x, xx = x
```

Idempotence is likely to appear early because it is easy to detect and common in important structures. Intersections and unions are idempotent. Logical conjunction and disjunction are idempotent. Projection-like and selection operations often are as well. The

oracle should distinguish: an idempotent element, an idempotent operation, and an idempotent transformation. For a transformation f:

```
f(f(x)) = f(x)
```

The same formal pattern reappears at different levels. An idempotent map behaves like a projection onto its image. Once an element has been transformed, applying the transformation again changes nothing.

This recurrence gives idempotence conceptual reach beyond finite binary tables. The machine may later discover that splitting idempotents creates new categorical objects. A property first encountered on diagonal entries of a table becomes a general organizing principle. This illustrates why a law's importance cannot be determined solely within the world where it is first found. Some elementary patterns become portals to advanced structure.

Absorbing elements and collapse

An element z is absorbing when:

```
zx = xz = z
```

for every x. Unlike an identity, which leaves other elements unchanged, an absorber erases them under combination. The identity preserves information:

```
ex = x
```

The absorber destroys it:

```
zx = z
```

These two roles define opposing operational tendencies. The oracle may classify elements by their effect on information: neutral, reversible, projective, absorbing, or expanding through combination. Such language is interpretive, but it may help the machine develop functional concepts rather than memorize equations. An absorber also affects cancellation. If z is absorbing and the carrier has more than one element, then:

```
zx = zy
```

for all x,y, while x and y need not be equal. Thus left cancellation fails whenever z acts on the left, unless the structure is trivial. The oracle may discover the implication:

```
two-sided absorber + nontrivial carrier ⇒ failure of cancellation
```

Here the plus sign remains inside the line rather than beginning it. This is a small example of theory interaction. One law constrains the possibility of another. The operation atlas can map such incompatibilities exhaustively.

Inverses as relational data

In a structure with an identity e, the oracle may search for elements y satisfying:

```
xy = e
```

or:

```
yx = e
```

These are right and left inverses of x. Again, the complete atlas reveals that the two notions need not coincide. In an associative structure with a two-sided identity, however, a left inverse and a right inverse must agree. Suppose:

```
yx = e
```

and:

```
xz = e
```

Then:

```
y = ye = y(xz) = (yx)z = ez = z
```

This proof uses associativity. The atlas may first suggest the pattern, but the proof identifies the precise hypothesis responsible. Without associativity, left and right inverses can behave differently. Thus the oracle learns to track theorem dependency. The conclusion does not follow from identity alone. It requires associativity.

This dependency is part of the theorem's content. The machine's library should store: premises, conclusion, proof, and countermodels obtained when each premise is removed. Such storage turns a theorem into a local map of logical necessity.

Cancellation and solvability

Cancellation laws concern the recoverability of inputs from outputs. Left cancellation:

```
xy = xz ⇒ y = z
```

means that fixing the left input x yields an injective transformation. For each x, define:

```
Lₓ(y) = xy
```

Then left cancellation is equivalent to every L_x being injective. Similarly, define:

```
Rₓ(y) = yx
```

Right cancellation means every R_x is injective. On a finite carrier, injective maps are automatically surjective. Therefore, in a finite cancellative system, the translations L_x and R_x are permutations. This creates a bridge from operation tables to permutation structure. The oracle can represent each row as a function from the second input to the output. If the row contains every carrier element exactly once, L_x is a permutation. Likewise, each column represents R_x. A table in which every row and every column is a permutation is a Latin square. The machine may discover this property geometrically before connecting it with equationsolving.

For every x and b, the equation:

```
xy = b
```

has a unique solution y. For every y and b, the equation:

```
xy = b
```

has a unique solution x. This solvability leads toward quasigroups and loops. The oracle need not be told those names. It can discover that a table with permutation rows and columns has a strong reversible character even without associativity. This is important because it prevents the machine from assuming that inverses and solvability

belong only to groups. There are alternative algebraic worlds. The complete atlas teaches plurality.

Associativity as a global coherence law

Many properties can be detected by inspecting individual rows, columns, or diagonal entries. Associativity differs. It compares nested uses of the operation across all triples. For each x,y,z, it requires:

```
(xy)z = x(yz)
```

The law relates two computation trees. One first combines x with y, then combines the result with z. The other first combines y with z, then combines x with the result. Associativity says that these local sequencing differences do not affect the final output. This is a coherence condition. Its significance grows with expression length. For four variables, there are five possible full parenthesizations: ((xy)z)w (x(yz))w (xy)(zw)

x((yz)w) x(y(zw)) Associativity proves that all five agree. For n inputs, the number of parenthesizations grows according to the Catalan numbers. A single law collapses this rapidly growing family into one unambiguous product. The compression becomes more dramatic at every length. This explains why associativity is likely to receive a high generativity score. It eliminates an entire dimension of syntactic bookkeeping. The oracle may observe that once associativity holds, it can write: x1 x2 ... xn without retaining parentheses. This notation is earned by proof. Associativity therefore does not merely classify a table. It creates a new language of finite products.

Commutativity as invariance under exchange

Commutativity requires:

```
xy = yx
```

It says that exchanging the two inputs leaves the output unchanged. This law is a symmetry under the transposition of input positions. Associativity concerns rebracketing. Commutativity concerns permutation. When both hold, any finite product becomes invariant under both parenthesization and ordering. The oracle can then represent a product by a multiset of factors rather than a sequence. This is another major compression. Without commutativity: xyz

and: zyx may differ. With commutativity and associativity, they agree. The machine can measure how many expression trees collapse under each law package. Associativity identifies bracketings while preserving sequence order. Commutativity identifies permutations while potentially retaining bracketings unless associativity is also present. Together they collapse expressions according to multiplicity alone. This offers a quantitative route to conceptual importance: measure the size of the equivalence classes generated in the free term algebra. A law that identifies many expressions provides substantial syntactic compression. But the machine must ensure that the quotient remains nontrivial and supports interesting models. An equation such as:

```
x = y
```

for all variables would collapse every element in every model, leaving only trivial structures. Maximum compression is not maximum value. A productive theory compresses while preserving a rich model class.

Implication mining

Once the atlas has been labeled by candidate properties, the oracle can search for implications. For equations E and F, it asks whether every tested model satisfying E also satisfies F: $\mathrm{Mod}_n(E) \subseteq \mathrm{Mod}_n(F)$ The subscript n emphasizes that the claim is restricted to carriers of size n. For several premises: Mod_n(E1 and E2 and ... and Ek) subset Mod_n(F). If the inclusion holds in the complete size-three atlas, the oracle records:

```
E₁,…,Eₖ ⇒ F verified for n = 3
```

It must not erase the size qualifier. Some implications hold only because finite carriers impose additional constraints. Others hold at size three but fail at size four. Still others are universally valid and admit symbolic proof. The system should therefore attempt three stages: finite verification, counterexample search at larger sizes, symbolic derivation. If symbolic proof succeeds, the status becomes universal relative to the adopted axioms and inference rules. If a counterexample appears, the implication is refuted. If neither occurs, it remains a bounded conjecture. This fail-closed discipline is essential. The oracle must prefer an unresolved label to an unjustified theorem.

Independence and minimal witnesses

Suppose a theory uses three laws: E_1, E_2, E_3 To determine whether E_3 is independent of E_1 and E_2, the oracle searches for a model satisfying: $E_1 \wedge E_2 \wedge \neg E_3$ A found model proves that E_3 cannot be derived from the first two. The smallest such model is especially valuable. It provides a minimal witness to independence. The finite atlas is therefore not only a source of examples. It is a source of logical boundary objects. A three-element countermodel may explain why a tempting proof cannot exist. The oracle can build an independence table showing, for each axiom, the smallest carrier size on which the remaining axioms hold while that axiom fails.

Such tables transform a list of axioms into a structural map of necessity. Human algebra texts often state that axioms are independent, but the concrete witnesses may be scattered or omitted. A verification-first oracle would treat them as part of the core record.

From tables to anonymous theories

After enough equations have been tested, the machine can group operations by the theories they satisfy. One anonymous theory may contain: E_7 E_{11} E_{24} Another may contain: E_7 E_{11} E_{24} E_{31} The second is a specialization of the first. The oracle can construct a directed graph in which an arrow indicates implication or extension by one independent law. This graph is more revealing than a flat list of named structures. It may show that several historically separate subjects share a common core. It may reveal alternative axiom paths leading to equivalent model classes. It may identify structurally central laws by the number of theory regions they connect.

The library begins to organize itself. At this point, the oracle still has not named semigroups, monoids, groups, bands, semilattices, quasigroups, or loops. It has discovered families of finite models and the law relations among them. Only an external comparison layer should attach human names. This separation creates a test of genuine reconstruction. If the anonymous theory lattice contains nodes corresponding closely to

familiar algebraic varieties, the machine has rediscovered part of the human classification from finite behavior.

If it organizes the landscape differently, that difference may also be informative. Human naming reflects historical interests. The oracle's organization may reflect computational centrality, compression, or structural connectivity. Neither organization should be assumed complete.

The limits of the three-element world

A complete three-element atlas is powerful, but it is not all of algebra. Some structures have no nontrivial models of size three. Some identities agree on every three-element operation but separate at larger sizes. Some phenomena require infinite carriers. Some definitions involve partial operations, relations, topology, order, grading, or multiple sorts. Some higher categorical structures cannot be represented faithfully by a single finite binary table. The atlas should therefore be treated as a seed environment, not a final universe. Its role is to teach the oracle several fundamental habits: distinguish law from accident, classify up to relabeling, seek counterexamples, preserve theorem status, identify compatible equivalences, construct quotients, and measure the generative effect of axioms.

These habits can later be transferred to richer worlds. The oracle may expand from one binary operation to: several operations, unary transformations, distinguished constants, relations,

typed carriers, families of maps, diagrams, local observations, and higher paths. The operation atlas provides the first complete arena in which the discovery cycle can be implemented without ambiguity.

The first algebraic cosmos

The three-element universe possesses a special philosophical appeal. It is not selected because three is mystical or because all mathematics is secretly ternary. It is selected because the entire space of binary operations is exactly enumerable and structurally nontrivial. The oracle can know that it has seen everything within the declared boundary. This completeness changes the nature of negative evidence. If no three-element table satisfies a proposed law package, then no such labeled operation has been missed. The theory has no model of that size. If every table satisfying E also satisfies F, the implication is genuinely exhaustive at that size.

The machine can therefore build a complete local map before moving into larger and less tractable spaces. This suggests a general strategy for mathematical ontogenesis: begin with worlds small enough to complete, extract invariant patterns, then transport those patterns into worlds too large to enumerate. Finite completeness supplies trustworthy seeds for infinite reasoning. The danger is overextension. A pattern found in every small world may fail later. The cure is not to abandon finite experimentation but to preserve domain labels and seek proofs. The complete atlas gives the oracle something humanity rarely possesses: a domain in which every case is available.

Within it, the machine can watch laws emerge as partitions of possibility, theories emerge as intersections of law classes, and abstract structures emerge as orbits under

relabeling. The raw universe is: all binary operations on A The first compression is: operations grouped by law profile The second is: operations grouped by isomorphism The third is: theories grouped by equivalent model classes The fourth is: theory implications organized into a lattice or graph At each step, the machine creates fewer but more meaningful objects. The direction is:

```
tables → invariants → classes → theories → architecture
```

This is the first visible birth of algebra. The next chapter examines the decisive abstraction more closely: how the oracle learns that different labels can conceal the same structure, and why isomorphism is not merely a convenient classification tool but a turning point in mathematical intelligence.

Chapter 7: Sameness Beneath Relabeling

Adapted from The Virgin Oracle, Chapter 5 (2026).

The complete atlas of finite operations initially presents the virgin oracle with a deceptive abundance. On a three-element carrier there are 19,683 labeled binary operation tables, and each table appears as a distinct sequence of nine outputs. If the machine classifies only by literal representation, it must treat all 19,683 as separate objects. But many of those differences are created solely by the names assigned to the three elements. Suppose one table uses the carrier:

```
A = {0, 1, 2}
```

and another uses:

```
B = {a, b, c}
```

The first operation may be written with numerals and the second with letters, yet the two tables may share exactly the same pattern. Replacing 0 by a, 1 by b, and 2 by c may transform one table into the other. The labels differ. The organization does not. This is the first setting in which the oracle must decide whether visible difference corresponds to mathematical difference. If it fails to make that distinction, its library will be filled with renamed copies. If it succeeds, it begins to classify structures rather than inscriptions. The transition can be stated simply: presentation ≠ structure A presentation is a particular encoding of an object. Structure is what remains invariant when the encoding is changed in an allowed way.

For finite operation tables, the allowed changes are permutations of the carrier. Let:

```
p : A → B
```

be a bijection. The map p is an isomorphism when it preserves the operation:

```
p(xy) = p(x)p(y)
```

for every x and y in A.

The multiplication on the left occurs in A. The multiplication on the right occurs in B. The same operation symbol may be used when context makes the distinction clear, but internally the oracle must preserve the typing. The equation says that two procedures agree. The first procedure is: combine x and y in A, then translate the result through p. The second is: translate x and y through p, then combine their images in B. When the procedures always agree, p preserves the operation. The square formed by these two routes commutes. The machine has encountered one of the central forms of abstract algebra:

```
transform before operating = operate before transforming
```

This pattern will later define homomorphisms, linear maps, functors, natural transformations, and many other structure-preserving processes. At the current stage, the transformation is also reversible. Therefore the two structures contain the same operational information. Write: A ~= B. The symbol ~= does not mean that A and B are

literally the same encoded object. It means that there exists a reversible operation-preserving translation between them. This distinction must remain explicit:

```
A = B means literal equality within the chosen representation.
```

A ~= B means equality of abstract structure up to reversible relabeling. The two judgments may coexist: A != B as tables, while A ~= B as algebras. There is no contradiction because the relations compare objects at different levels.

The action of relabeling

For a carrier with three elements, there are six possible permutations. Each permutation rearranges the labels while preserving their distinctness. If the carrier is:

```
A = {0,1,2}
```

the permutations include the identity, three swaps, and two cyclic rearrangements. Each permutation acts on every operation table. Suppose the original operation is · and p is a permutation. Define the relabeled operation $\cdot_p$ by:

```
x ·ₚ y = p⁻¹(p(x) · p(y))
```

This equation transports the original operation through p. It ensures that p becomes an isomorphism from the relabeled table to the original one:

```
p(x ·ₚ y) = p(x) · p(y)
```

The six permutations therefore produce up to six labeled presentations of the same abstract operation. Not every orbit has six distinct tables. If some nontrivial permutation preserves the table exactly, two or more relabelings may produce the same presentation. Such a permutation is an automorphism. An automorphism is an isomorphism from a structure to itself:

```
p : A → A
```

with:

```
p(xy) = p(x)p(y)
```

The automorphisms of A form a group under composition:

Aut(A)

The appearance of this group is significant. The machine began by trying to eliminate redundant labels. In doing so, it discovered the internal symmetry of each operation. The amount of redundancy is controlled by symmetry. If an operation has no nontrivial automorphisms, its orbit under relabeling has six distinct tables.

If it has several automorphisms, its orbit is smaller because some relabelings leave it unchanged. The relation is:

```
|Orbit(A)| × |Aut(A)| = 6
```

for a three-element carrier. More generally:

```
|Orbit(A)| × |Stab(A)| = |Sₙ|
```

where S_n is the permutation group of the n-element carrier and Stab(A) is the stabilizer of the operation table under relabeling. Because the stabilizer is precisely the automorphism group:

```
|Orbit(A)| × |Aut(A)| = n!
```

The oracle may discover this pattern experimentally before knowing orbit-stabilizer theory. It observes that highly symmetric tables have fewer distinct labeled copies, while asymmetric tables have more. Classification by isomorphism therefore reveals automorphism groups automatically. Redundancy becomes evidence of symmetry.

Canonical representatives

For computation, the oracle needs a practical way to store one representative from each isomorphism class. One method is to encode each operation table as a sequence and select the lexicographically smallest sequence among all of its relabelings. Define:

```
can(A) = min{p·A | p ∈ Sₙ}
```

Here p·A denotes the table obtained by relabeling A through p. Then:

```
A ≅ B ⇔ can(A) = can(B)
```

The canonical representative is not intrinsically more mathematical than the other presentations. It is merely selected by a deterministic convention.

This distinction is important. Canonicalization should not be confused with discovering a naturally preferred labeling. The operation itself may contain no reason to call one element 0 rather than 1. The canonical labeling is imposed for storage and comparison. The machine should preserve: the canonical table, the original table, the permutation connecting them, and the full automorphism group. If it stores only the canonical table, it gains efficiency but loses provenance. It can no longer explain how an encountered presentation was recognized as a known structure. A verification-first system should be able to report: This table belongs to isomorphism class S_{42}.

The canonical representative is C_{42}. The relabeling p witnesses the isomorphism. The table has automorphism group $\mathrm{Aut}(C_{42})$. The witness p matters. Classification should not become an unsupported assertion of sameness.

Equality classes and isomorphism classes

At first glance, isomorphism classes resemble ordinary equivalence classes. The The relation ~= is reflexive: A ~= A because the identity map preserves every operation. It is symmetric:

```
A ≅ B ⇒ B ≅ A
```

because the inverse of an isomorphism is also an isomorphism. It is transitive:

```
A ≅ B ∧ B ≅ C ⇒ A ≅ C
```

because the composition of isomorphisms is an isomorphism. Thus isomorphism is an equivalence relation on the collection of finite operation systems. The oracle can form the quotient: Structures / ~= . Each point in this quotient represents one abstract isomorphism class. But the quotient alone does not retain the individual isomorphisms. It says only whether two presentations belong to the same class. This loss is harmless for some counting problems. If the goal is merely to determine how many abstract operation types exist, the set of isomorphism classes may suffice. For other questions, the lost maps are essential. Two structures may be connected by several different isomorphisms. A

structure may possess many automorphisms. Those transformations reveal internal symmetry and influence later constructions.

If every isomorphism class is collapsed to a featureless point, that information disappears. The machine therefore faces two possible abstractions. The coarse abstraction is the quotient set: Structures / ~= . The richer abstraction retains structures as objects and isomorphisms as arrows. This forms a groupoid. In a groupoid, every arrow is reversible. Between two objects there may be zero, one, or many arrows. Each object has an automorphism group formed by its loops. The action groupoid of the relabeling problem contains: operation tables as objects, permutations preserving operations as arrows. The quotient set records the connected components of this groupoid.

The groupoid records the paths within each component. The distinction can be summarized as:

```
quotient set = which objects are equivalent
groupoid = how the objects are equivalent
```

This is the first place where the virgin oracle must decide whether the witness of sameness is itself mathematical data. If it preserves witnesses, it opens a route toward higher structure.

Symmetry as self-equivalence

An automorphism is a structure-preserving relabeling that returns a structure to itself. It reveals that the object has multiple presentations even after the underlying carrier is fixed. Consider an operation with a distinguished identity e. Any automorphism must preserve e because the identity is characterized uniquely by its behavior. If p is an automorphism, then p(e) is also an identity:

```
p(e)p(x) = p(ex) = p(x)
```

and:

```
p(x)p(e) = p(xe) = p(x)
```

Because the identity is unique:

```
p(e) = e
```

Thus a structural role constrains symmetry. Similarly, an absorbing element, when unique, must be fixed by every automorphism. Sets of idempotents are carried to sets of idempotents. Elements of a given order are carried to elements of the same order in group-like structures. The machine can therefore discover invariants by studying what automorphisms preserve. A property P is structural when:

```
P(x) ⇒ P(p(x))
```

for every automorphism p. The more refined statement is that structural properties are invariant under every isomorphism, not merely automorphisms. If f : A → B is an isomorphism and x has a role in A, then f(x) has the corresponding role in B. This supplies a criterion for whether a discovered feature depends on labels.

Suppose the machine identifies "the element named 0" as special. That feature is not invariant under arbitrary relabeling. Suppose instead it identifies "the unique two-sided identity." That role is invariant. The oracle should prefer invariant descriptions. This

declared irrelevant by the theory. If the theory ignores carrier labels, then meaningful properties must be preserved under relabeling. If the theory ignores coordinate systems, meaningful properties must be coordinate invariant. If the theory ignores categorical presentation, meaningful properties should be invariant under equivalence.

The choice of equivalence determines the admissible language of properties.

The danger of canonical labels

Canonicalization is computationally useful but conceptually dangerous. Once the oracle assigns a canonical table to an isomorphism class, it may begin treating the canonical labels as if they were structural. It may say, for example, that element 0 is always the identity because its canonicalization procedure tends to place the identity first. This would confuse a representational convention with a theorem. The identity is not structurally the element labeled 0. It is the element satisfying:

```
ex = xe = x
```

Canonicalization may choose to rename that element 0, but the order of explanation must remain clear: behavior determines the role, then the convention assigns the label. Not: the label determines the role.

This principle has broad importance for machine-discovered mathematics. Automated systems often rely on canonical encodings, sorted forms, normalized expressions, and selected representatives. These are necessary for avoiding duplication, but they can create artifacts. A machine may find a pattern in the canonicalization procedure rather than in the mathematical structures. Every discovered conjecture should therefore be tested for invariance under changes of representation. If a claimed law disappears when the canonicalization convention changes, it is probably not structural. The oracle needs an audit layer that asks: Does this statement refer only to invariant features?

Does it survive relabeling? Does it depend on the chosen representative? Can it be reformulated without privileged names? This is a form of theory-preserving discipline.

Isomorphism as compression

Suppose an isomorphism class has six labeled presentations. Storing all six operation tables independently requires six times as much table data. Instead, the oracle can store: one canonical table, the permutations producing its presentations, and its automorphism group. This is already a compression. But the deeper gain is conceptual. Every theorem proved using only structural properties transfers automatically to every isomorphic copy. If A ~= B and A satisfies an equation E, then B also satisfies E.

```
For an equational law s = t, the proof follows because an isomorphism preserves every
```

operation used to evaluate s and t. Thus:

```
A |= E and A ~= B  =>  B |= E
```

The law profile is constant on isomorphism classes. The machine no longer needs to test every labeled copy independently once the isomorphism class has been identified. More importantly, explanations become portable. A proof about the canonical representative is not merely a fact about one chosen table. It is a fact about the entire

abstract structure. The isomorphism acts as a transport mechanism. Compression and transport are linked. The oracle can spend effort once and reuse the result across all presentations.

From isomorphism to invariant

An invariant assigns the same value to isomorphic structures. Let I be a function on structures. It is an isomorphism invariant when:

```
```

Examples in the finite operation world include: the number of idempotent elements, the existence of an identity, the number of absorbing elements, the size of the automorphism group, the law signature, the number of subalgebras, and the multiset of sizes of congruence classes. An invariant can help prove that two structures are not isomorphic. If:

I(A) != I(B) implies A and B are not isomorphic. Equality of one invariant does not usually prove isomorphism. Different structures may share the same number of idempotents or the same automorphism-group size. The oracle therefore develops families of invariants. A coarse invariant partitions the atlas into large blocks. Adding more invariants refines the partition. A complete invariant would satisfy:

```
I(A) = I(B) ⇔ A ≅ B
```

The canonical form is a complete invariant, but it may be expensive to compute in larger settings. Much of classification theory involves finding effective invariants that capture enough structure without solving the entire isomorphism problem directly. This introduces another form of mathematical intelligence: choosing useful invariants. An invariant is valuable when it is: easy to compute, stable under the relevant equivalence, strong enough to distinguish many structures, and connected to deeper theory. The oracle may learn to construct invariants systematically from operations and maps. For each element x, it may examine the transformations:

```
Lₓ(y) = xy
Rₓ(y) = yx
```

It may record ranks, cycle structures, fixed points, or images of these transformations. The resulting multisets are preserved under relabeling. It may construct a directed graph from the operation table and apply graph invariants. It may count solutions to equations:

```
N_E(A) = |{v | A satisfies E under assignment v}|
```

The number of satisfying assignments is invariant under isomorphism. Thus even an equation that is not universally valid can produce a quantitative invariant. Instead of asking only whether:

```
(xy)z = x(yz)
```

holds for all triples, the oracle may count how many triples satisfy it. Define the associativity score:

```
α(A) = |{(x,y,z) ∈ A³ | (xy)z = x(yz)}|
```

An associative operation has:

```
α(A) = |A|³
```

Nonassociative operations may have intermediate scores. This creates a graded landscape rather than a binary partition. The same method applies to commutativity:

```
κ(A) = |{(x,y) ∈ A² | xy = yx}|
```

and idempotence:

```
ι(A) = |{x ∈ A | xx = x}|
```

These scores are structural invariants. They may help the oracle detect near-laws, deformations, and pathways between theories. A structure need not satisfy an equation universally for the equation to reveal its organization.

Approximate structure and exact structure

Human abstract algebra usually treats equations exactly. An operation is associative or it is not. A map is a homomorphism or it is not. A discovery engine may benefit from tracking near-satisfaction during exploration. Suppose one operation violates associativity on only one of 27 triples. Another violates it on 20 triples. Both are nonassociative, but the first lies closer to the associative region. This notion of distance may guide conjecture and repair. The oracle might ask:

Can one table entry be changed to produce associativity? Which associative structures are nearest under Hamming distance? Does a near-associative table possess other group-like features? Are there paths through operation space along which laws appear in a recognizable order? These questions create a geometry of finite algebraic structures. However, approximate satisfaction must never be confused with exact law. A table with one associativity defect is still not associative. The machine should maintain two layers: exact classification, exploratory proximity. The first supports theorem and proof. The second supports discovery and search. This distinction may be especially useful for nascent intelligence. Exact laws form sharp boundaries, but near-laws may reveal which boundaries are structurally accessible.

Isotopy and alternative notions of sameness

Isomorphism uses one bijection p to relabel both inputs and the output:

```
p(xy) = p(x)p(y)
```

But other transformations may preserve a weaker or different structure. Suppose three bijections f, g, and h satisfy:

```
h(xy) = f(x) ·′ g(y)
```

This relation is called an isotopy between binary operations. The left input, right input, and output may be relabeled independently. Two operations can be isotopic without being isomorphic. A virgin oracle exploring transformation-based compression might discover such relations before human terminology is introduced. It may notice that certain operation tables become identical when rows, columns, and symbols are permuted separately. Which notion of sameness should it use?

There is no universally correct answer. Isomorphism preserves one kind of structure. Isotopy preserves another. Homotopy equivalence, Morita equivalence, and categorical equivalence preserve still others. The appropriate relation depends on which

experiments and constructions the theory regards as meaningful. The oracle should therefore avoid declaring one final notion of identity. It should construct a hierarchy of equivalences. For finite binary operations, possible levels include: literal table equality, equality after simultaneous relabeling, equality after independent row, column, and output relabeling, agreement on a bounded equation language, equivalence of associated categories or representations.

Each quotient reveals one pattern and conceals another. The machine must state the equivalence relation whenever it classifies. The phrase "these are the same" is incomplete without: the same under what transformations?

Structural identity as an evolving decision

The notion of sameness changes as the oracle's language expands. At the earliest stage, two states are the same when their stored encodings match. After operation is introduced, two elements may be observationally equivalent when no available term distinguishes them. After homomorphisms are introduced, structures may be considered the same up to isomorphism. After categories are introduced, categories may be considered the same up to equivalence rather than strict isomorphism. After homotopy is introduced, spaces may be considered the same up to homotopy equivalence.

After univalence, equivalence may induce identity inside a suitable universe. The sequence is not merely a weakening of equality. It is an enrichment. Literal equality gives one relation. Isomorphism gives explicit reversible maps. Category equivalence includes functors and natural isomorphisms. Homotopy equivalence includes paths and higher deformations. Univalent identity retains equivalence as identity data within type theory. The oracle's growing mathematics is therefore partly a history of increasingly adequate identity criteria. At each stage, it discovers that its previous notion of sameness was too rigid or too forgetful. Too rigid means that equivalent presentations remain unnecessarily separate.

Too forgetful means that collapsing them erases relevant transformations. The correct abstraction preserves exactly the structure needed for the next stage.

Skeletons and the temptation to collapse

In category theory, a skeleton selects one object from each isomorphism class. This resembles choosing canonical representatives for finite operations. A skeleton can remove redundancy. But it does not necessarily preserve every feature of the original category literally. It preserves the category only up to equivalence. The operation atlas already contains the seed of this issue. A set of canonical representatives is smaller than the full groupoid of labeled tables. Yet the groupoid contains explicit relabelings and automorphisms that the representative set alone does not display. One may reconstruct some of that information by attaching Aut(A) to each representative, but this still does not always replace the full network naturally.

The machine should therefore maintain both: a skeleton for efficient indexing, and a groupoid for structural provenance. This dual representation mirrors a broader principle:

canonical representatives support computation, equivalence witnesses support mathematics. Neither should replace the other entirely.

The first moduli space

After quotienting the operation atlas by isomorphism, the oracle obtains a collection of abstract operation types. But a mere list does not capture how those types relate. The machine can organize them by: shared equations, subalgebra relations, quotients, homomorphisms, deformations of table entries, automorphism groups, and inclusion among theories. This organized collection resembles a finite moduli space: a space whose points represent structures up to isomorphism, enriched by symmetry and relation data. A moduli problem asks: What are all structures of a given kind, up to an appropriate equivalence? The complete three-element operation atlas is therefore a primitive moduli problem.

The objects are all binary operations on a three-element carrier. The equivalences are relabelings. The automorphism groups record stabilizers. The quotient gives isomorphism classes. The groupoid retains symmetry. The law profiles stratify the space. The theory graph organizes the strata.

This vocabulary belongs to advanced mathematics, but its underlying architecture emerges naturally from finite classification. The machine begins with a database of tables and ends with an anonymous moduli atlas. That transition is conceptually significant. It suggests that moduli need not be introduced as a remote geometric abstraction. Moduli begin whenever an intelligence asks for all structures of a kind while refusing to count renamed copies as different.

The discovery of structural objecthood

Before isomorphism, the oracle treats a structure as a table. After isomorphism, it treats the table as one presentation of something more abstract. What is that abstract thing? One answer is the isomorphism class. Another is the entire groupoid of presentations and isomorphisms. Another is the object as characterized by its relations to all other objects. No single answer is sufficient in every foundational framework. But the machine has discovered a crucial fact: the mathematical object is not exhausted by the symbols used to display it. This discovery changes the status of notation. Symbols become coordinates on structure rather than structure itself.

It also changes the meaning of knowledge. To know an operation is no longer only to know its table. It may mean knowing its laws, symmetries, substructures, quotients, representations, and universal properties. The object becomes a node in a network. This prepares the oracle for homomorphisms. Isomorphisms preserve all structure reversibly, but many important maps preserve structure without being reversible. Such maps reveal images, kernels, collapses, embeddings, and quotients. Before studying those maps, however, the oracle must understand a more general form of compatible sameness inside a single structure. Two elements may be different yet become indistinguishable under a quotient. The relation identifying them must respect every operation.

That relation is a congruence. The next chapter examines how congruence turns equivalence into construction and how quotienting creates genuinely new algebraic worlds.

Chapter 8: Congruence, Quotient, and the Birth of Structure

Adapted from The Virgin Oracle, Chapter 6 (2026).

The virgin oracle has learned that two operation tables may differ only by relabeling. Isomorphism removes distinctions between entire structures when a reversible translation preserves their operations. A second and equally important form of abstraction occurs inside a single structure. Two elements may be different, yet the oracle may decide that their difference is irrelevant for a particular purpose. It may identify them and attempt to treat their equivalence classes as new elements. But an arbitrary identification can destroy the operation. The machine therefore needs a criterion separating legitimate quotienting from indiscriminate collapse.

That criterion is congruence.

From equivalence to congruence

Let A carry a binary operation ·, and let ~ be an equivalence relation on A. Because ~ is reflexive, symmetric, and transitive, it partitions A into equivalence classes:

```
[x] = {y ∈ A | y ~ x}
```

The quotient set is:

```
A/~ = {[x] | x ∈ A}
```

The oracle would like to define an operation on the quotient by:

```
[x][y] = [xy]
```

This looks natural. Choose one representative from each class, combine them in A, and take the class of the result. The difficulty is that a class may contain several representatives. Suppose: x ~ x′ and: y ~ y′

The quotient product is meaningful only if: xy ~ x′y′ Otherwise, choosing x and y produces one output class while choosing x′ and y′ produces another. The supposed operation on equivalence classes would not be well defined. The required compatibility condition is:

```
x ~ x′ ∧ y ~ y′ ⇒ xy ~ x′y′
```

An equivalence relation satisfying this condition is a congruence. The distinction is fundamental: equivalence organizes elements into classes congruence permits those classes to inherit operations An equivalence relation says which elements are to count as alike. A congruence says that the operations cannot detect the difference.

Operational indistinguishability

Congruence can be understood experimentally. Suppose $x \sim x'$. If the relation is to represent genuine operational indistinguishability, replacing x by x' inside any permitted calculation should produce an equivalent result. For a binary operation: $xy \sim x'y$ and: $yx \sim yx'$ for every y. Applying this repeatedly gives compatibility with every term built from the operation. If: $x_i \sim y_i$ for each variable position, then: $t(x_1,\ldots,x_n) \sim t(y_1,\ldots,y_n)$ for every term t.

Thus congruence is not merely compatibility with one immediate multiplication. It propagates through the entire grammar of the algebra. The oracle can test this in finite structures. It begins with a proposed partition of the carrier and asks whether every operation respects the blocks. For each pair of blocks B and C, all products xy with $x \in B$ and $y \in C$ must land in one quotient block. If the products land in several blocks, the partition is incompatible. The machine can therefore build a quotient table only when each pair of input classes has a uniquely determined output class. This produces a concrete algorithm: choose a partition of A test operation compatibility retain the partition if compatibility holds construct the induced operation on the blocks The resulting quotient may contain fewer elements than the original structure while preserving a controlled image of its operation.

The smallest and largest congruences

Every algebra possesses at least two congruences. The equality congruence identifies only elements that are literally equal:

```
x ~ y ⇔ x = y
```

Its quotient changes nothing: A/Delta ~= A. At the other extreme, the universal congruence identifies every pair: $x \sim y$ for all $x,y \in A$ Its quotient contains one element. These two congruences define the outer limits of abstraction. The equality congruence forgets nothing.

The universal congruence forgets everything about individual elements. Between them may lie intermediate congruences that preserve some structure while collapsing other distinctions. The machine's task is not to maximize collapse. It is to find quotients that reveal useful organization. A quotient is informative when it simplifies the structure without reducing it to triviality.

Congruence closure

The oracle may begin not with a complete equivalence relation but with a small set of desired identifications. Suppose it wishes to impose: $a \sim b$ This single relation may force additional identifications. Because the relation must be symmetric: $b \sim a$ Because it must be transitive, any elements already related to a or b may become related to each other. Because it must respect the operation: $ax \sim bx$ and: $xa \sim xb$ for every x. Those new identifications may generate still more through repeated multiplication. The smallest congruence containing the original relation is its congruence closure. Write:

Cg(a,b)

for the least congruence identifying a and b. More generally, for a relation R:

```
Cg(R) = intersection of all congruences containing R
```

The process resembles deductive closure. A small collection of proposed identifications expands until it becomes stable under equivalence and operation. The closure laws are again: $R \subseteq Cg(R)$

```
R ⊆ S ⇒ Cg(R) ⊆ Cg(S)
Cg(Cg(R)) = Cg(R)
```

The same algebraic pattern has reappeared. Earlier, equations generated deductive closure among terms. Now, element identifications generate congruence closure inside a model. In fact, these are two views of the same construction. Equations imposed on a free algebra generate a congruence, and the quotient by that congruence produces the algebra presented by those equations.

Generators and relations

Let T(X) be a free term algebra on generators X. Suppose the oracle chooses a set of equations:

```
E = {sᵢ = tᵢ}
```

Each equation asks that the terms s_i and t_i become identical in the new structure. The equations generate a congruence $\equiv E$ on T(X). The quotient is:

```
A = T(X)/≡E
```

This construction is written:

```
A = ⟨X | E⟩
```

The notation means that A is generated by X subject to the relations E. The free term algebra represents unrestricted symbolic construction. The congruence records the equations imposed.

The quotient turns the imposed equations into literal equality of equivalence classes. Thus:

```
formal equation before quotient → identity after quotient
```

If $s \equiv E\ t$, then:

```
[s] = [t]
```

inside A. The oracle has created a world in which the proposed relations are true by construction. This is not merely observing a preexisting operation table. It is mathematical world-building.

Quotient as controlled forgetting

A quotient removes distinctions, but it does so according to a rule. Suppose A contains elements that differ only by some feature the oracle no longer wishes to track. A congruence identifies precisely those elements while preserving the operation visible at the coarser level. The quotient map is:

```
q : A → A/~
```

defined by:

```
q(x) = [x]
```

The map q is structure preserving:

```
q(xy) = q(x)q(y)
```

because:

```
q(xy) = [xy]
```

and:

```
q(x)q(y) = [x][y] = [xy]
```

The quotient map translates detailed elements into coarse classes without disrupting the operation. This is why quotienting is more than deletion. It is a homomorphism from a richer structure to a coarser one.

The details lost by q are exactly the distinctions within each class. The details preserved are those visible through the quotient operation.

Kernels create congruences

Suppose f is a homomorphism:

```
f : A → B
```

with:

```
f(xy) = f(x)f(y)
```

The map may send distinct elements of A to the same element of B. Define:

```
x ~f y ⇔ f(x) = f(y)
```

This relation is an equivalence relation. It is also a congruence. If:

```
f(x) = f(x′)
```

and:

```
f(y) = f(y′)
```

then:

```
f(xy) = f(x)f(y)
```

and:

```
f(x′y′) = f(x′)f(y′)
```

Therefore:

```
f(xy) = f(x′y′)
```

so: xy ~f x′y′ The relation ~f is the kernel congruence of f. It records exactly which distinctions the map erases. Two source elements lie in the same kernel class when the target cannot distinguish them.

This gives a deep operational interpretation: a congruence is a pattern of distinctions invisible to some structure-preserving observation The quotient by the kernel congruence reproduces the part of A visible through f.

The first isomorphism pattern

```
Let f : A → B be a homomorphism.
```

The image of f is:

```
im f = {f(x) | x ∈ A}
```

The quotient by the kernel congruence is: A/~f Define a map:

$\bar{f} : A/\sim f \to \text{im } f$

by:

$\bar{f}([x]) = f(x)$

This map is well defined because elements in the same class have the same image. It is injective because:

$\bar{f}([x]) = \bar{f}([y])$

implies:

```
f(x) = f(y)
```

which implies:

```
[x] = [y]
```

It is surjective by definition and preserves the operation. Therefore A/~f ≅ im f.

This is the general finite-algebra form of the first isomorphism theorem. The equation expresses a recurring architecture:

source / distinctions erased by the map ≅ visible image. The pattern appears throughout algebra. For groups: G/ker f ≅ im f. For rings: R/ker f ≅ im f. For modules: M/ker f ≅ im f. The exact meaning of kernel changes with the structure, but the architecture persists. The machine may first discover the relation by enumerating finite homomorphisms. It notices that the number of kernel classes equals the number of image elements and that the induced operation tables agree up to relabeling. It can then derive the theorem abstractly. This is an ideal example of autonomous mathematical development: finite observation suggests a pattern the pattern is formulated anonymously counterexamples are sought a general proof explains the recurrence the result is transported across many algebraic theories

A theorem becomes fundamental when the same architecture survives changes of

interpretation.

Normal subgroups as congruences

In a group, a congruence can be encoded by a special subgroup. Let ~ be a group congruence and let e be the identity. Define:

```
N = {x ∈ G | x ~ e}
```

This is the equivalence class of the identity. The entire congruence can be reconstructed from N:

$x \sim y \Leftrightarrow x^{-1}y \in N$

The subgroup N must be normal:

$gNg^{-1} = N$

Normality is exactly the condition needed for coset multiplication to be well defined:

```
(xN)(yN) = xyN
```

Thus the apparently special human definition of a normal subgroup arises naturally from the general requirement that an equivalence relation respect multiplication. A virgin oracle could discover normality not as an isolated property, but as the internal signature of quotientability. The sequence is: compatible equivalence on G identity class N normality of N quotient group G/N The concept of normal subgroup is therefore downstream of congruence and quotient. This order may be more conceptually natural than introducing normality as a technical condition without explaining why it is needed.

Ideals as congruence data

A similar phenomenon occurs in rings. Let I be the class of 0 under a ring congruence. Then:

```
x ~ y ⇔ x - y ∈ I
```

For the relation to respect multiplication, I must absorb multiplication by arbitrary ring elements: rI ⊆ I and, in a noncommutative ring: Ir ⊆ I

Such a subset is an ideal. The quotient ring consists of cosets:

```
R/I = {x + I | x ∈ R}
```

with operations:

```
(x + I) + (y + I) = (x + y) + I
```

and:

```
(x + I)(y + I) = xy + I
```

The ideal is not simply a subset selected for historical reasons. It is precisely the kind of subset that can serve as the zero class of a ring congruence. Again, quotientability determines the concept. For modules, submodules play the same role. For universal algebras more generally, congruences replace normal subgroups and ideals as the universal language of quotient structure. This suggests that a nascent intelligence might discover congruence before separating the human subjects of group theory, ring theory, and module theory. The specialized objects would appear as local forms of one general construction.

The congruence lattice

The set of all congruences on A can be ordered by inclusion. If θ and φ are congruences, write: θ ≤ φ when every pair identified by θ is also identified by φ. Then θ is finer than φ because it makes fewer identifications. The equality congruence is the least element. The universal congruence is the greatest. The intersection of any collection of congruences is again a congruence. This gives their meet:

```
θ ∧ φ = θ ∩ φ
```

The join is the smallest congruence containing both:

```
θ ∨ φ = Cg(θ ∪ φ)
```

Thus the congruences of A form a lattice:

Con(A)

The lattice records every legitimate way of collapsing A. A path upward in Con(A) means forgetting more distinctions. A path downward means recovering finer distinctions. This turns quotienting into a navigable geometry of abstraction. The oracle can examine which congruences cover which others, which quotient sizes occur, and how congruence structure correlates with other properties. Some algebras have only the two extreme congruences. They are simple. For a simple algebra, every homomorphism is either injective or has a one-element image, subject to the surrounding algebraic setting. The term simple therefore describes resistance to nontrivial quotienting.

The machine might discover simplicity by observing that no intermediate compatible partition exists. This gives a structural interpretation:

```
simple object = object with no nontrivial internal collapse compatible with its operations
```

Successive quotients

Suppose $\theta \leq \varphi$ are congruences on A. The quotient A/θ still carries a congruence induced by φ. Quotienting again gives a structure corresponding to A/φ. Schematically: $(A/\theta)/(\varphi/\theta) \cong A/\varphi$. This is a quotient-of-a-quotient principle. The oracle learns that
compatible collapses can be performed in stages. First identify according to θ. Then identify the remaining distinctions specified by φ.

The result is equivalent to performing the coarser identification directly. This gives quotienting a compositional structure. The machine can factor a large abstraction into smaller, interpretable steps. Such factorizations may reveal hidden layers within a theory. A complicated quotient can be decomposed into elementary collapses, each of which has a clear operational meaning.

Factorization of homomorphisms

Every homomorphism f can be factored through its quotient by the kernel: $A \longrightarrow q\ A/{\sim f} \to \bar{f} \to \text{im}\ f \to i \to B$. Here q is the quotient map, $\bar{f}$ is an isomorphism, and i is the inclusion of the
image into B Thus:

```
f = i ∘ f̄ ∘ q
```

The map f performs three conceptually distinct acts: it collapses indistinguishable source elements it renames the resulting classes as image elements it includes that image into the larger target This factorization exposes the anatomy of a structure-preserving map. The quotient part explains information loss. The isomorphism part preserves structure exactly. The inclusion part places the result inside a larger world. A machine that stores only input-output behavior may miss this architecture. A mathematically intelligent oracle should seek canonical factorizations that explain how a map operates. This pattern later generalizes to categorical factorization systems.

The universal property of a quotient

A quotient is not characterized only by its equivalence classes.

Let $q : A \to A/\theta$ be the quotient map. Suppose $f : A \to B$ is a homomorphism that identifies every pair related by θ:

```
x θ y ⇒ f(x) = f(y)
```

Then there exists a unique homomorphism:

$$\bar{f} : A/\theta \to B$$

such that:

$$f = \bar{f} \circ q$$

The map $\bar{f}$ is defined by:

$$\bar{f}([x]) = f(x)$$

The condition on f guarantees that this definition does not depend on the representative. The universal property says: Every structure-preserving map that ignores the distinctions collapsed by θ must pass uniquely through the quotient. This is stronger than saying the quotient has equivalence classes. It characterizes the quotient by its relationship to all compatible maps. The quotient is the most general structure in which the chosen identifications have become equalities. This idea is central to advanced mathematics. A construction is often best understood not by how its elements were assembled, but by which maps factor uniquely through it.

The virgin oracle has encountered another universal property.

Coequalizing two maps

A congruence may be generated by demanding that two maps agree. Suppose:

```
f,g : X → A
The oracle wants a quotient q : A → Q satisfying:
q ∘ f = q ∘ g
```

This means that for every $x \in X$:

```
q(f(x)) = q(g(x))
```

Thus the quotient identifies f(x) with g(x). The smallest congruence accomplishing this is generated by all pairs: (f(x),g(x)) The resulting quotient is a coequalizer of f and g. Its universal property is: If $h : A \to B$ satisfies $h \circ f = h \circ g$, then there exists a unique $\bar{h} : Q \to B$ with:

$$h = \bar{h} \circ q$$

The coequalizer is the universal way to force two maps to become equal. This formulation shifts attention from element pairs to parallel arrows. The oracle is moving toward category theory. At the elementary level, it says: identify these specified outputs At the categorical level, it says: construct the universal object in which two arrows agree The same quotient process has been lifted from elements to diagrams.

Quotienting syntax

The machine's term algebra also carries congruences. Suppose the oracle adopts associativity:

```
(xy)z = x(yz)
```

This equation generates a congruence on all binary terms. Every two parenthesizations of the same ordered sequence become equivalent. For four variables: ((xy)z)w (x(yz))w

(xy)(zw) x((yz)w) x(y(zw)) all belong to one congruence class. The quotient of the free binary term algebra by associativity is the free semigroup-like structure on the variables. Its elements can be represented by nonempty words: x1 x2 ... xn. Parentheses have disappeared because the quotient has made them irrelevant. If commutativity is also imposed, variable order becomes irrelevant. The quotient elements can be represented by multisets or exponent vectors. If idempotence is added:

```
xx = x
```

repeated occurrences may collapse as well. Each equation package produces a different normal form and a different quotient universe. Thus algebraic theories can be understood as designs for forgetting syntactic distinctions. Associativity forgets bracketing. Commutativity forgets ordering. Idempotence forgets multiplicity beyond presence. Identity laws forget inserted neutral elements. Inverse laws permit cancellation of opposite factors. The equations specify which features of an expression survive into the quotient.

Normal forms as chosen representatives

A quotient class may contain many terms. For calculation, the oracle often selects a canonical representative called a normal form. Under associativity, a normal form may be a flat word. Under associativity and commutativity, it may be a sorted word.

Under group laws, it may be a reduced word with adjacent inverse pairs removed. The normal form provides an efficient representation of the quotient element. But it must not be confused with the quotient element itself. The class is the mathematical object. The normal form is a chosen presentation. This is the same distinction encountered in canonical labeling of operation tables. Again: canonical form supports computation equivalence class carries the abstraction The machine should preserve a proof that each term reduces to its normal form and that two terms have the same normal form exactly when they are equivalent. Without those guarantees, normalization is merely a heuristic.

Quotienting and information loss

Every nontrivial quotient loses information.

```
If q(x) = q(y), the quotient cannot recover whether the source was x or y.
```

This loss may be intentional. The quotient was designed to ignore precisely that distinction. But the oracle should quantify what has been lost. For a finite quotient, it can record the sizes of the fibers: $q^{-1}([x])$ Large fibers indicate substantial collapse. It can preserve the kernel congruence and the quotient map so that the abstraction remains traceable. This produces a layered record: source structure A congruence θ quotient structure A/θ quotient map q

fiber data proof of operation compatibility No quotient should appear as an unexplained replacement of one structure by another. The oracle's policy should be: forget operationally, remember historically The quotient operates at the coarser level, but provenance remains available.

When quotienting destroys too much

A compatible quotient is mathematically legitimate, but it may still be unhelpful. The universal congruence always produces a valid one-element quotient. Yet this quotient may erase every distinction relevant to the investigation. The machine needs a criterion for useful quotienting. A quotient may be valuable when it: reveals a simpler recurring structure isolates an invariant image turns a partial pattern into an exact law supports factorization of maps separates essential from inessential degrees of freedom or exposes a hierarchy of substructures A quotient may be unhelpful when it: collapses nearly all models to the same trivial object erases symmetries required for later analysis loses witnesses of equivalence or depends on an arbitrary partition without explanatory value The oracle must therefore evaluate quotients by what they preserve and enable, not merely by how much they compress.

Quotient sets, groupoids, and higher quotients

When the oracle forms A/~ as a set of classes, it forgets the internal paths connecting representatives. Suppose several transformations witness x ~ y. The quotient class records only that x and y are identified. A groupoid quotient retains the witnesses as arrows. A higher quotient may also retain transformations among those arrows. This distinction matters when symmetry is part of the object. Consider quotienting a set by a group action. The ordinary orbit set records only the orbits. Points with different stabilizer groups may become indistinguishable at the level of the orbit set. The action groupoid retains those stabilizers as automorphisms.

In geometric and topological settings, this richer quotient can preserve singularity and symmetry information lost by the set-theoretic quotient. The virgin oracle should therefore learn that quotient is not one operation but a family of constructions at different levels of retained coherence. The hierarchy is: quotient set: retain classes quotient groupoid: retain equivalence arrows higher quotient: retain arrows, transformations among arrows, and higher coherence The correct level depends on what the theory intends to remember.

Effective equivalence

An equivalence relation may be presented abstractly, but a quotient map produces a specific relation:

```
x ~ y ⇔ q(x) = q(y)
```

An equivalence relation is effective when it arises as the kernel relation of its quotient map. In ordinary sets, every equivalence relation is effective. In more general categories, this need not be automatic. The relationship between equivalence relations and quotients becomes a structural property of the ambient category.

This issue later becomes important in exact categories and topoi. A topos supports well-behaved quotients of internal equivalence relations. The elementary act of grouping finite elements into compatible classes eventually becomes part of the architecture of a mathematical universe. The development is continuous: finite partitions algebraic congruences kernel pairs coequalizers effective equivalence relations quotients in topoi

higher quotients A basic operation of classification becomes a central organizing principle of advanced mathematics.

Quotienting as theory formation

The machine's own knowledge can also be viewed as a quotient. It begins with many expressions, presentations, proofs, and models. It identifies expressions related by accepted equations. It identifies structures related by isomorphism. It identifies theories generating the same closure. It identifies constructions satisfying the same universal property. Each identification replaces a larger collection with a more invariant object. The oracle's library is therefore built through successive quotients: raw strings / grammar terms / equations models / isomorphism theories / equivalent closure

categories / equivalence types / univalent identification At every stage, the quotient criterion becomes more sophisticated. The danger is circularity. The machine uses its current theory to decide which distinctions are irrelevant, then builds a new theory from the quotient. A mistaken equivalence could distort every later stage. For this reason, quotient formation must be fail-closed and reversible at the level of provenance. The machine may adopt a quotient for computation while retaining the unquotiented source and the witness data needed to audit the choice. Mathematical development then becomes a sequence of explicit commitments rather than silent erasures.

The birth of a new object

The deepest lesson of quotienting is that identification can create rather than merely reduce. Before the quotient, [x] does not exist as an element of the original carrier. It is a collection of source elements tied together by a relation. After the quotient, [x] is one element of a new structure. The operation on classes may possess properties not visible in the same form at the source level. A complicated structure may map onto a simpler cyclic or commutative quotient. A nontrivial kernel may isolate the obstruction to injectivity. A quotient by a commutator-like congruence may produce the largest commutative image. New algebraic objects are born from disciplined collapse.

This suggests a general ontogenetic step:

$$A_{n+1} = T(A_n)/\equiv_n$$

The machine first generates constructions $T(A_n)$. It then identifies them according to a congruence $\equiv_n$. The quotient becomes the next carrier of thought. The process is neither pure expansion nor pure reduction. It is expansion followed by structural compression.

The oracle's quotient protocol

A verification-first virgin oracle should never form a quotient casually. It should record the following information. First, the source structure and all operations being preserved. Second, the proposed generating identifications. Third, the resulting congruence closure. Fourth, a proof that each operation respects the relation. Fifth, the induced quotient operations. Sixth, the quotient map. Seventh, the universal factorization property. Eighth, the information lost in each fiber. Ninth, the equivalence witnesses retained outside the quotient. Tenth, the constructions newly enabled by the quotient. This protocol transforms quotienting from a hidden simplification into an auditable mathematical event.

Structure through loss

The word structure often suggests additional organization: more operations, more relations, more axioms. Quotienting reveals the opposite movement. Structure can also emerge by removing distinctions. A mass of terms becomes an algebra when equations identify redundant constructions. A labeled atlas becomes a moduli space when relabelings are ignored. A homomorphism becomes intelligible when its kernel collapse is separated from its image. Local presentations become one global object when overlap distinctions are reconciled. Equivalent types become identifiable when univalence treats equivalence as identity. The creation of structure therefore alternates between enrichment and forgetting.

The oracle adds operations to expose behavior.

It removes distinctions to expose invariance. Neither process is sufficient alone. An intelligence that only generates becomes overwhelmed by possibilities. An intelligence that only quotients collapses everything into triviality. Mathematics inhabits the productive tension between them. The central pattern of this chapter is:

```
equivalence + operation compatibility → congruence
congruence + quotient map → new algebra
homomorphism + kernel congruence → image factorization
generators + relations → presented structure
local identifications + retained provenance → controlled abstraction
```

The quotient is the point at which recognized sameness becomes an object-producing operation. The next chapter follows one particularly important path through the finite theory landscape. Without supplying its human name, the oracle will combine associativity, identity, and reversible action and discover a family of structures equivalent to groups.

Chapter 9: Algebraic Abiogenesis: The Operation-First Reversal

Adapted from Mathematical It from Bit (2026), Sections 1-2.

The conceptual story can now be made formal. Algebraic Abiogenesis begins with finite carriers and anonymous operations, evaluates formal terms exactly, and treats stable equivalence as evidence from which mathematical objects can be constructed.

1. From Axiomatic Algebra to Algebraic Abiogenesis

1.1 The Conventional Top-Down Origin of Algebraic Systems

Abstract algebra is ordinarily presented as the study of already constituted mathematical species. A group is introduced as a set equipped with a binary operation satisfying associativity, identity, and inverse axioms. A ring is supplied with two operations and a prescribed interaction between them. A lattice begins with meet and join, a Boolean algebra adds complement and distinguished bounds, and a field is defined by an interlocking system of additive and multiplicative laws. Once the signature and axioms have been chosen, mathematical work proceeds by determining consequences, classifying models, constructing homomorphisms, studying congruences, identifying free objects, and comparing varieties.

This mode of development is top-down. The investigator supplies the ontology before beginning the investigation. The primitive operations are named, their intended functions are understood, and the class of admissible structures is delimited by explicit laws. Even when alternative axiom systems are sought, the intended algebraic species is usually fixed in advance. The task is to simplify, strengthen, weaken, or reorganize a theory whose conceptual identity is already known. Automated theorem proving largely inherits this arrangement. A prover receives a formal language, axioms, definitions, and target statements. Automated conjecture generation relaxes the requirement that the target theorem be specified, but it normally operates within a supplied domain. Equation-discovery systems may search for relations among known functions, yet the functions themselves are already selected and interpreted. Finite-model finders construct structures satisfying or refuting stated formulas, but the formulas are given before the

models are sought. In each case, mathematical discovery begins after the principal ontological decisions have been made. This leaves a prior problem comparatively unexplored. Before one proves theorems about a group, how could the idea of a group arise? Before one writes lattice axioms, how might repeated operational regularities suggest that meet and join constitute a coherent species? More generally, can a collection of anonymous operations generate the evidence from which its own algebraic theory is inferred? Algebraic Abiogenesis addresses this earlier stage. It does not initially ask what follows from a selected axiom system. It asks how an axiom system, a model class, and eventually an abstract algebraic object might emerge from finite operational behavior without a named target.

1.2 The Operation-First Reversal

The methodological reversal begins with finite carriers and anonymous operation tables. An anonymous operation has a known arity and a complete finite interpretation, but no assigned semantic role. A binary table is not called addition, multiplication, composition, conjunction, meet, or implication. A ternary table is not assumed to be a median, discriminator, majority operation, or conditional. The operation is treated only as a finite mapping whose values can be composed into formal terms. From this minimal substrate, the system generates expressions and evaluates them on all assignments. Whenever two distinct terms induce exactly the same finite function, an observational identity is recorded. A single collision may be accidental, but recurrent collisions across independently generated algebras and multiple carrier orders can indicate a transferable regularity. Families of such regularities are compressed into provisional bases, tested against countermodels, and examined for their ability to support a larger deductive structure.

The resulting direction of development is

anonymous finite operations

```
→ exact term-function equivalences
→ transferable identities
→ provisional axiom systems
→ varieties or related model classes
→ structural mathematics.
```

The procedure is not designed to recover a predetermined catalogue entry. It is not rewarded specifically for associativity, commutativity, distributivity, identities, inverses, absorptions, or cancellation. Indeed, such familiar laws may arise, but they receive no privileged status merely because they are recognizable. The primary selection criteria concern algebraic fertility: nondegenerate coordinate dependence, exact recurrent regularity, explanatory compression, deductive reserve, resistance to counterexamples, and the capacity to support representations, normal forms, free objects, or other structural developments. This reversal also changes the role of finite models. In conventional model finding, a finite algebra is usually sought because it satisfies a known theory or refutes a proposed consequence. In Algebraic Abiogenesis, finite algebras initially precede the theory. They are not merely models of equations; they are sources of evidence from which candidate equations are induced. The theory is born from recurring invariances in the operations themselves.

The method therefore distinguishes operation generation from equation generation. Directly enumerating short identities and asking which have models explores a space of laws. Algebraic Abiogenesis instead explores a space of operational worlds and asks which laws those worlds

naturally express. The difference is analogous to proposing a grammar and finding sentences that satisfy it versus observing utterances and inferring the grammar that organizes them.

1.3 Mathematical It from Bit

The phrase It from Bit expresses the idea that stable entities may arise from informational distinctions and their organization. In the present mathematical setting, the primitive bits are literal or effectively finite: entries in operation tables, values of terms on assignments, equality decisions, finite partitions, and finite records of recurrence. The emergent "its" are not physical objects but mathematical objects: equivalence classes of expressions, quotient term algebras, model classes, free structures, congruence systems, and decidable equational theories. Let a formal term be regarded initially as a syntactic tree. Two terms may look entirely different while behaving identically throughout a finite ecology of algebras. When the system identifies them under every relevant assignment, it treats their difference as observationally invisible relative to that ecology. Repeated identification induces an equivalence relation on formal expressions. Because term evaluation respects composition, this equivalence is a congruence.

The quotient by that congruence is therefore an algebraic object whose elements are not raw expressions but stable classes of expressions. In this sense, mathematical objecthood appears through the controlled loss of distinction. A quotient element exists because many syntactic presentations have become interchangeable under the available operations. The object is not inserted into the system as a primitive name. It is formed by the invariances of the observational regime. The It-from-Bit interpretation does not imply that finite data uniquely determine a final infinite theory. Any induction from bounded observations requires policies governing recurrence, simplicity, explanatory power, and counterexample response. Different search pressures can reveal different structures within the same operational space. The method therefore does not remove mathematical judgment; it formalizes part of the process by which judgment can be applied reproducibly.

Nor does the analogy reduce mathematics to arbitrary data compression. A short identity is valuable only if it survives independent models, explains a substantial body of observations, supports exact deduction, and remains meaningful under homomorphic images, subalgebras, products, or other relevant constructions. Algebraic birth requires more than numerical regularity. It requires the emergence of lawful substitution and stable structural consequences. The central proposal is consequently precise: abstract mathematical objects can be generated as quotient structures induced by recurrent finite operational equivalence. Finite information supplies the distinctions; exact invariance determines which distinctions may be discarded; quotient formation creates the objects.

1.4 Scope and Principal Claims

This paper presents Algebraic Abiogenesis as a general methodology rather than as a claim about one historically new equation. The research object is the process by which anonymous finite operations are converted into coherent algebraic theories. The method contains several stages. It generates anonymous finite algebras, enumerates bounded formal terms, detects exact term-function collisions, and identifies recurrent theories across carrier orders. It rejects constants, projections, operation aliases, definitional collapses, and theories whose apparent richness consists only of shallow consequences. It subtracts identities already explained by provisional laws and searches the remaining residual theory.

Candidate bases are attacked with adversarial countermodels, tested for independence, and promoted only when they support transferable structure. Two completed experimental cycles establish feasibility. The first recovered a known binaryoperation theory, thereby serving as a calibration and exposing the need for stronger novelty and collapse controls. The second applied the revised method to an unrelated ternary signature, rejected a misleading large collision family after explanation subtraction, isolated a recurring residual theory, and completed that theory through rewriting, normal forms, free algebras, and exact finite-model enumeration. These experiments do not yet establish that the method will routinely produce previously unknown algebra. They establish a more limited but essential result: a signature-independent, operation-first process can begin with finite anonymous tables and end with a mathematically mature equational theory without receiving the final theory as a target.

The principal claim is therefore methodological. Algebraic Abiogenesis provides a reproducible framework for studying the birth, selection, refinement, rejection, and completion of abstract algebraic systems. Its success should ultimately be measured not by the number of equations generated, but by whether it repeatedly transforms minimal finite information into coherent, extensible, independently auditable mathematics.

2. Formal Foundations: Stable Quotients from Finite Operational Data

2.1 Signatures and Absolutely Free Term Algebras

Algebraic Abiogenesis begins with a finitary signature rather than with a named algebraic species. A signature τ is a collection of operation symbols together with their arities. It may contain one binary symbol, one ternary symbol, several operations of different arities, constants, or any other finite combination suitable for the experiment. At this stage the symbols

possess no intended mathematical interpretation. They are placeholders for operations whose laws, if any, are to be discovered. Let X be a nonempty set of variables. The absolutely free term algebra $T\tau(X)$ consists of all formal expressions that can be built from X using the operation symbols in τ. If τ contains one binary symbol , then variables are terms and, whenever s and t are terms, the expression st is a term. If τ contains one ternary symbol m, then m(r,s,t) is a term whenever r,s,t are terms. The construction continues recursively without assuming any identities. The term algebra records syntax

before mathematical equivalence has been introduced. Two terms are distinct whenever they are distinct formal trees. Thus (xy)z and x(yz) are initially different, as are m(x,y,y) and x.

No associativity, commutativity, idempotence, absorption, or other law is built into the grammar. The term algebra is therefore maximally discriminating. It preserves every distinction created by variable placement, operation symbol, and parenthesization. This absolute freedom is essential. If the term generator were to simplify expressions using familiar algebraic laws, the system would import the very structure it is intended to discover. Algebraic Abiogenesis therefore treats all laws as empirical or deductive outcomes rather than syntactic conventions. The term algebra also provides a universal domain in which discoveries from different finite models can be compared. Every finite τ-algebra interprets the same formal terms.

Consequently, recurrent identities can be expressed as equivalences inside one common syntactic universe, even when the underlying finite carriers and operation tables differ. For computation, $T_\tau(X)$ is explored through finite approximations. Terms may be bounded by size, depth, variable count, operation count, or another structural complexity measure. Let $T_{d\tau}(X)$ denote a finite term universe selected by such a bound d. The bounded universe is not itself the mathematical theory. It is the observational window through which the provisional theory is inferred.

A useful term representation preserves the complete rooted tree. Variables are leaf nodes, while applications of primitive operations are internal nodes labeled by operation symbols. Canonical serialization permits exact hashing, comparison, and storage. Symmetries such as variable renaming may later be recognized for theory compression, but they are not identified at the initial syntactic stage. The formal distinction between syntax and semantics will drive the entire method. Syntax supplies the space of possible expressions. Finite anonymous algebras supply meanings for those expressions. Stable agreement among meanings then induces equivalence classes that were absent from the original term algebra.

2.2 Anonymous Finite Algebras

A finite τ-algebra A consists of a finite carrier set together with one concrete operation for every symbol in τ. If f is an n-ary operation symbol, its interpretation is a function

$$f^A : A^n \to A.$$

For a carrier of order k, the complete interpretation of f can be stored as a table with k^n entries. A binary operation requires k^2 entries, while a ternary operation requires k^3 entries. The word anonymous does not mean that the operation lacks arity, notation, or a finite implementation. It means that no mathematical role is assigned to it in advance. A binary operation is not presumed to be addition, multiplication, composition, meet, or implication. A ternary operation is not presumed to be a median, majority, discriminator, affine combination, or conditional operator. The system receives only the table and the formal permission to compose it.

An anonymous algebra may be generated randomly, evolved under a fertility objective, constructed by constrained synthesis, or obtained through mutation of another

operation table. The generation mechanism affects which regions of operation space are explored, but it does not alter the definition of the finite algebra itself. For a signature with operation symbols f1,…,fr, a finite algebra may be encoded as

```
A = (A,f1A,…,frA).
```

The carrier labels are arbitrary. Two tables related by a permutation of the carrier represent isomorphic algebras. During early search, labeled tables may be retained because they are computationally convenient. Later stages can quotient by isomorphism to avoid overcounting repeated presentations of the same structure. The anonymous setting separates extensional data from intensional interpretation. Extensionally, every output of every operation is known. Intensionally, no law explains why

those outputs are organized as they are. Algebraic Abiogenesis attempts to construct that missing intensional layer. Finite operation tables are suitable primitive substrates for several reasons. They support exact evaluation, exhaustive assignment testing, complete congruence enumeration at small orders, exact subalgebra and quotient construction, and finite countermodel generation. They also eliminate numerical approximation. An identity either holds on every relevant assignment or it does not. At the same time, finite tables are epistemically incomplete. A finite algebra may satisfy identities that fail in nearby or larger structures. A single table therefore cannot determine a general equational theory. The method treats every finite algebra as a theory-bearing observation rather than as final proof of a universal law.

A collection E of finite τ-algebras will be called an algebraic ecology. Its members may differ in carrier order, table structure, and discovery history. The purpose of the ecology is to support identities that recur across multiple independently generated operational worlds while exposing identities that are peculiar to one table. The ecology is dynamic. New models may be added because they were generated independently, because they satisfy a provisional basis, or because they were synthesized adversarially to falsify a candidate consequence. As the ecology expands, the observed theory typically becomes smaller and more discriminating.

2.3 Term-Evaluation Maps

Every formal term becomes a finite function when interpreted in a finite algebra. Let A be a finite τ-algebra and let X be a finite variable set. An assignment is a function

```
α : X → A.
```

Each term t ∈ Tτ(X) has a value tA(α) ∈ A, defined recursively. If t is a variable x, then

```
tA(α) = α(x).
```

If

```
t = f(t1,…,tn),
```

then tA(α) fA(t1A(α),…,tnA(α)).

The term therefore induces a function

```
tA : AX → A,
```

where AX is the set of all assignments from X to A. When X contains r variables and |A|=k, the complete term function contains k^r output values.

All term functions can be collected into an evaluation map

```
φA : Tτ(X) → A^(AX),
where A^(AX) denotes the set of functions AX → A.
```

The evaluation map is a homomorphism from the absolutely free term algebra into the algebra of term functions under pointwise operations. If f is n-ary, its pointwise interpretation on functions g1,...,gn is given by f(g1,...,gn)(α) fA(g1(α),...,gn(α)). Thus evaluation respects all formal compositions. For computational purposes, a term function may be stored as a finite vector. Fix an ordering α1,...,αN

```
of all assignments X → A. Then t is represented by
```

[tA(α1),...,tA(αN)]. Two terms induce the same term function precisely when these vectors are identical entry by entry. No statistical test or approximate distance is required. Exact equality of finite vectors provides an exact certificate that the corresponding identity holds in A. This complete evaluation is one of the foundational strengths of the method. Evolutionary search may use approximate agreement as a gradient, but final identity recognition is exact. A candidate law is never promoted because it is usually true, numerically close, or highly correlated. It must agree on every assignment in every model used for certification. The evaluation map also connects finite computation to universal algebra. The set of identities satisfied by A is exactly the kernel relation of φA. Term-function collision is therefore not merely a heuristic pattern. It is the computational manifestation of an equational congruence.

In large searches, complete evaluation can become expensive. The number of assignments grows exponentially with the number of variables, while the number of terms grows rapidly with depth. Several optimizations are available: vectorized evaluation, memoization of term

functions, hash-consing of terms, elimination of duplicate functions, and retention of one shortest representative for each semantic class. The shortest-representative strategy is especially important. Once a term function has been encountered, later terms producing the same function need not be retained as new semantic objects. Instead, each later term is recorded as an identity with the current representative. This converts uncontrolled syntactic growth into a finite catalogue of semantic equivalence classes for each bounded search.

2.4 Observational Congruences

Define a relation ΘA on Tτ(X) by s ΘA t if and only if

```
sA = tA
as functions AX → A.
```

Equivalently,

s ΘA t exactly when

```
sA(α)=tA(α)
for every assignment α : X → A.
```

Because ΘA is the kernel of a homomorphism, it is a congruence on Tτ(X). It is reflexive, symmetric, transitive, and compatible with every operation symbol in τ. If si ΘA ti

```
for i=1,…,n, then
```

f(s1,…,sn) ΘA f(t1,…,tn). This compatibility is what turns repeated finite agreement into algebraic structure. The system is not merely clustering expressions by resemblance. It is identifying expressions in a way that remains stable under all formal contexts. Now let E be an ecology of finite τ-algebras. Define ΘE

$\Theta E = \bigcap \{ \Theta A : A \in E \}$.

Thus

s ΘE t exactly when s and t induce the same term function in every algebra in E. An intersection of congruences is again a congruence, so ΘE is a congruence on Tτ(X). It represents the complete equational theory jointly observed in the ecology. Under the usual interpretation of identities as universally quantified over variables, ΘE is also fully invariant. Let σ be an endomorphism of Tτ(X), corresponding to an arbitrary substitution of terms for variables. If s ΘE t,

```
then every algebra in E satisfies s=t under every assignment. Replacing variables by
arbitrary
```

terms preserves validity, so σ(s) ΘE σ(t). Full invariance is crucial because an equational theory must remain stable under substitution. Without it, observed collisions would not support general algebraic reasoning. The congruence ΘE can be regarded as an observational identity policy. Two formal expressions are identified when the ecology cannot distinguish them by any assignment. The more diverse the ecology, the stronger the test of indistinguishability. For finite computation, one usually works with a bounded restriction of ΘE to Tdτ(X). This restricted relation records which bounded terms remain equal throughout the ecology. It approximates the full congruence but does not automatically reveal all consequences involving larger terms.

The methodology therefore distinguishes three levels:

1. exact finite validity inside one algebra;

2. exact recurrence throughout a finite ecology;

3. universal derivability from an extracted basis.

The first two are observational. The third is deductive. Algebraic Abiogenesis moves from the observational congruence toward a compact deductive representation of it.

2.5 Emergent Quotient Objects

Once ΘE has been defined, one may form the quotient algebra Tτ(X)/ΘE.

Its elements are equivalence classes

[t]E

```
{u ∈ Tτ(X) : u θE t}.
```

Operations are induced by

f([t1]E,…,[tn]E) [f(t1,…,tn)]E. This definition is well-defined because ΘE is a congruence. The quotient is the first precise sense in which a mathematical object emerges from the finite operational ecology. Before quotienting, every formal tree is a separate syntactic object. After quotienting, expressions that remain indistinguishable throughout the ecology become multiple presentations of one algebraic element. The quotient may be finite or infinite. Even when every member of E is finite, the common quotient theory may admit infinitely many inequivalent terms. Conversely, a strongly collapsing ecology may produce a small quotient. The quotient should not automatically be identified with the free algebra of a final variety. It is initially the free algebra of the observational theory represented by ΘE. If a basis Σ is later extracted and proved to generate exactly ΘE, then Tτ(X)/ΘE can be identified with the free algebra of Mod(Σ) on X.

In bounded computation, one obtains a finite semantic quotient of the explored term universe. Its classes are represented by distinct combined term-function fingerprints across the ecology. These classes serve as provisional objects.

```
The identities observed in E are compressed statements about this quotient. A law s=t
asserts
```

that [s]E and [t]E are the same object. A basis is a small set of such identifications from which a larger portion of ΘE can be generated. This perspective reverses the customary order of explanation. In axiomatic mathematics, equations are supplied and quotient structure follows. Here, quotient behavior is observed first

through exact finite semantics, and equations are selected afterward as concise explanations of that behavior. The quotient interpretation also clarifies why raw collision count is not an adequate fertility measure. A single shallow law can generate thousands of collisions. Those collisions may describe only one simple quotienting mechanism. A fertile theory should exhibit a quotient structure whose regularities cannot be exhausted immediately by one trivial explanation. Algebraic Abiogenesis therefore studies not only how many distinctions disappear, but how the identifications are organized, how economically they can be described, and what structural mathematics follows from them.

2.6 Refinement by Countermodels

Suppose E and E′ are algebraic ecologies with $E \subseteq E'$.

Then

$\Theta E' = \bigcap \{ \Theta A : A \in E' \} \subseteq \Theta E = \bigcap \{ \Theta A : A \in E \}$.

Therefore,

ΘE′ ⊆ ΘE. Adding models can preserve an observed identity or destroy it, but cannot create a new identity that was absent from the smaller ecology. This monotonicity is one of the central formal properties of the method. The development of a theory is a refinement process. Early ecologies often identify too many terms because they contain too few operational perspectives. Additional models restore distinctions that were accidentally erased. If s ΘE t but s is not ΘE′-equivalent to t, then some newly added algebra separates them. That

```
algebra acts as a countermodel to the provisional identity s=t.
```

The quotient maps move in the opposite direction. Since ΘE′ is finer than ΘE, there is a natural surjective homomorphism

Tτ(X)/ΘE′

```
→
```

Tτ(X)/ΘE. The refined quotient contains more distinct expression classes and maps onto the coarser earlier quotient by forgetting the newly recovered distinctions. Thus theory development can be viewed as increasing conceptual resolution. A coarse ecology produces large equivalence classes. Countermodels split those classes, yielding a more discriminating ontology. This process provides a mathematical interpretation of adversarial theory formation. Candidate identities are not merely tested after discovery. They are actively challenged by searches for algebras satisfying the current basis while falsifying proposed consequences. Successful countermodels are added to the ecology, forcing ΘE to refine.

The refinement sequence may be written

ΘE0 ⊇ ΘE1 ⊇ ΘE2 ⊇ …, where E0 ⊆ E1 ⊆ E2 ⊆ … . At each stage, the method may recompute semantic classes, extract a revised basis, and measure the unexplained residual theory. There is no guarantee that a finite sequence of ecologies determines a unique ultimate theory. The search depends on which countermodels are generated and which operational regions are explored. The methodology therefore records the entire refinement history rather than presenting the final basis as though it had appeared without selection. A mature theory candidate should survive several kinds of refinement:

- independently generated models;
- models of new carrier orders;
- random models constrained by the provisional basis;
- adversarial models targeting specific consequences;
- quotients, subalgebras, and products when computationally feasible;
- alternative search seeds and term bounds.

Survival under refinement does not alone prove universal validity. Universal validity follows from deduction once a basis has been established. But refinement determines whether the proposed basis is a plausible explanation of a transferable algebraic phenomenon rather than a description of a narrow finite sample.

The formal sequence

finite operational data

```
→ observational congruence
→ quotient term algebra
→ countermodel refinement
→ compact equational explanation
```

constitutes the foundational mechanism of Algebraic Abiogenesis. It is the precise mathematical core of the claim that abstract algebraic objects can emerge from finite anonymous information.

Chapter 10: The Discovery Cycle and the Rejection of False Births

Adapted from Mathematical It from Bit (2026), Sections 3-4.

3. The Algebraic Abiogenesis Discovery Cycle

3.1 Generation of Anonymous Operations

The discovery cycle begins with finite carriers and anonymous primitive operations. Let τ be a fixed finitary signature. For each carrier size n, a candidate τ-algebra is represented by one complete finite table for every operation symbol in τ. A binary operation contributes n^2 entries, a ternary operation contributes n^3 entries, and a nullary operation contributes one distinguished carrier element. The initial population may be generated uniformly at random, but purely random sampling is rarely efficient. Most finite operation tables lie near one of two unproductive extremes. Some behave chaotically and support almost no short transferable identities. Others collapse rapidly to constants, projections, aliases, or extremely small images and therefore exhibit large numbers of formally true but structurally uninformative equations. Algebraic Abiogenesis searches between these extremes.

Candidate algebras may be created or modified by entry mutation, row or column replacement, block copying, carrier relabeling, recombination of operation tables, and crossover between parent structures. For higher-arity operations, analogous slices or subtables may be exchanged.

No mutation operator contains a preferred algebraic identity. A mutation changes extensional operational behavior without declaring what kind of law should result. The generation process is signature-independent. The same architecture can operate on one binary operation, one ternary operation, several binary operations, mixed unary-binary signatures, or signatures containing constants. What changes is the size of the tables, the term grammar, and the computational cost of exhaustive evaluation. Carrier relabeling must be treated carefully. Relabeling produces an isomorphic table and therefore no new algebraic structure. Such permutations can nevertheless be useful as mutation operators because they disrupt superficial table correlations while preserving the underlying model. During diversity measurement and final enumeration, isomorphic duplicates should be detected or discounted.

The operation generator is not expected to produce a mature theory directly. Its role is to create an ecology rich enough for transferable semantic regularities to appear. A useful population should contain operational diversity, multiple carrier orders, and enough structural continuity that related identity fingerprints can recur without requiring identical tables.

3.2 Term Generation and Exact Collision Detection

For a chosen variable set X and complexity bound d, the system generates a finite term universe Tdτ(X). Terms may be ordered by tree size, depth, number of operation nodes, or a weighted complexity measure adapted to the signature. Breadth-first generation is generally preferable because it discovers short explanations before long ones. Each newly formed term is serialized canonically so that syntactic duplicates are removed. At this stage, no algebraic simplification is applied. Terms remain distinct unless their formal trees are identical. For every candidate algebra A and every term t, the complete term function

```
tA : AX → A
```

is evaluated. The resulting finite vector is used as a semantic key. If two distinct terms s and t produce the same key, then

```
A |= s=t.
```

The equality is exact because every assignment has been examined. A semantic dictionary retains one preferred representative for each term function. The representative is ordinarily the shortest term, with lexicographic or structural tie-breaking.

When another term evaluates to the same function, the pair is recorded as an observational identity rather than retained as a new semantic class. This procedure converts a rapidly growing syntactic term set into a smaller collection of exact semantic classes. It also produces an identity catalogue containing the equations witnessed by the finite model. Not every collision is equally informative. Several filtering stages may be applied. Identities between two variable-renamed copies of the same trivial form may be marked as symmetry variants. Collisions involving constant term functions may be separated from those retaining dependence on multiple variables. Identities in which both sides are already explained by a shorter known law may be deferred to the residual-theory stage.

Approximate agreement may be useful during evolutionary search. For terms s and t, one may calculate the fraction of assignments on which they agree. This gives a gradient that can favor mutations approaching exact structure. Approximate agreement, however, never constitutes an accepted law. Promotion requires complete equality of the relevant term-function vectors. The collision engine is therefore both semantic and exact. It does not infer laws from numerical correlation. It computes the kernel of finite term evaluation within the chosen observational window.

3.3 Cross-Order Recurrence

An identity satisfied by one finite algebra may result from a peculiar coincidence of its operation table. Even identities appearing in several algebras of the same small order may be artifacts of that order. Algebraic Abiogenesis therefore promotes recurrence across independently generated models and, whenever possible, across different carrier sizes. Let E1,...,Ek be independently generated model families, possibly of orders n1,...,nk. A candidate identity is transferable at the observational level when it holds exactly in a specified fraction of these models and appears in more than one carrier order. The strongest initial requirement is universal recurrence throughout a selected ecology:

```
A |= s=t
```

for every A in E. For exploratory work, weaker thresholds may be used to detect clusters of related theories. An identity might recur in one coherent subpopulation while failing elsewhere. This does not necessarily make it unimportant. It may indicate that the population contains several competing theory embryos.

Cross-order recurrence serves several purposes. It reduces accidental promotion, discourages theories tied to a single cardinality, and supplies evidence that the law reflects a structural mechanism rather than a specific table pattern. Independence of model generation is equally important. Ten minor mutations of one parent table do not provide the same evidential strength as ten separately generated models. The research record should preserve lineage, random seed, carrier order, and search condition so that recurrence can be interpreted correctly. The method may also require recurrence under constrained resynthesis. Once a provisional basis Σ has been extracted, new models are generated subject to Σ but without copying the original tables. If the larger observed theory reappears, this supports the claim that Σ captures a genuine generative mechanism.

Cross-order recurrence does not prove an equation universally. Its function is theory selection. Universal validity is later obtained by formal derivation from an accepted basis.

3.4 Promotion from Observations to Theory Candidates

The discovery cycle distinguishes finite observations from mathematical theories. An equation recorded in a finite table is an exact fact about that table. A recurring equation is an exact fact about an ecology. Neither is automatically a theorem about all algebras satisfying some yetundetermined basis. Promotion occurs only when a coherent family of identities exhibits several properties simultaneously. First, the identities must recur across independent models and preferably across carrier orders. Second, they must involve nondegenerate term functions rather than only constants or projections. Third, they should admit compression by a substantially smaller set of explanatory laws. Fourth, the explanatory laws should generate a reserve of additional exact consequences.

Fifth, the resulting theory should survive attempts to synthesize countermodels. A provisional theory candidate consists of more than a list of equations. It includes:

- a signature;
- a certified model ecology;

- a bounded term universe;
- an observed identity relation;
- a proposed basis;
- an explanation map from observed identities to basis consequences;
- residual identities not yet explained;
- countermodels to rejected alternatives;
- structural invariants of the supporting models.

Promotion may occur at several levels. A weak candidate may be retained as a finite regularity. A stronger candidate may be designated a transferable equational embryo. A mature candidate may be treated as a proposed variety once a basis has been shown to explain the desired theory and formal structural analysis has begun. The method should also permit negative outcomes. A stable identity family may be rejected because it is already known, because one operation is definable from another, because all consequences reduce to a shallow law, or because adversarial synthesis destroys the apparent reserve. Rejection is part of the discovery cycle rather than evidence that the cycle failed.

3.5 Basis Extraction

A large collision catalogue is not itself an intelligible theory. The next task is to find a small set of identities that explains as much of the observed congruence as possible. Basis extraction begins by ranking candidate equations. Shorter identities are usually preferred, but length alone is insufficient. A useful law should merge many observed term classes under bounded equational closure. It should also survive cross-order testing and avoid dependence on accidental constants or carrier labels. A greedy compressor may proceed as follows. Start with an empty basis. At each step, select the identity that explains the largest weighted portion of the unexplained observed theory. Add it to the provisional basis, recompute bounded consequences, and repeat until the residual theory falls below a threshold or no candidate produces meaningful compression.

More advanced extraction may use minimum-description-length objectives, integer optimization, proof search, or counterexample-guided minimization. The objective is not merely to minimize the number of axioms. A slightly larger basis may be preferable if it yields simpler proofs, a clearer structural interpretation, or a convergent rewrite orientation. Variable renamings, dual laws, direct substitutions, and trivial transitive consequences should be grouped before basis selection. Otherwise, the compressor may waste several basis slots on equivalent presentations of one mechanism. Irredundancy is tested after extraction. For each basis identity σ, the system searches for a finite algebra satisfying every identity in Σ except σ while falsifying σ. A successful countermodel

certifies that σ is not a consequence of the remaining basis. Failure to find a small countermodel does not prove dependence, but it identifies the identities requiring symbolic analysis. The basis extracted from bounded observations remains provisional until its universal consequences are understood. The compressor may select laws that explain the observed term universe but fail to generate longer identities seen at greater depth. Consequently, basis extraction and residual search must be repeated as the term bound and model ecology expand.

3.6 Structural Maturation

A theory candidate becomes mathematically significant only when its identity basis supports a broader structural development. Algebraic Abiogenesis therefore continues beyond basis discovery. The first maturation task is often representation. The defining identities may reveal a canonical unary map, a family of retractions, a block decomposition, a translation action, a centralizer condition, or another internal structure through which the primitive operations factor. Such a representation can convert opaque equations into transparent model data. The second task is the construction of free algebras. Terms are reduced modulo the proposed theory, and canonical representatives are sought. A successful construction shows what the theory creates without accidental identifications imposed by finite models.

The third task is rewriting. Basis identities are oriented when possible into terminating reduction rules. Critical overlaps, nonlinear substitutions, and context interactions are analyzed. A convergent rewrite system yields unique normal forms and a decision procedure for the equational word problem. The fourth task is finite model theory. The system enumerates labeled models, identifies isomorphism types, computes congruences, subalgebras, quotients, simple members, and subdirectly irreducible members. Exact counting formulas are preferred to brute-force totals when structural classifications permit them. The fifth task is countermodel compilation. If two terms have different canonical forms, the method attempts to construct a finite algebra and assignment separating them. This converts nonderivability into an explicit certificate.

The sixth task is literature and equivalence analysis. A candidate theory must be compared not only by its displayed equations but also under operation renaming, duality, opposite multiplication, definitional extension, term equivalence, reducts, expansions, and alternative axiom systems. A rediscovery is retained as a calibration result but not presented as a new variety.

Structural maturation may reveal that the original candidate was mischaracterized. One operation may be auxiliary, one law may imply several others, or the theory may belong to a known class. The methodology permits such reinterpretation. The purpose is not to defend the first description but to reach the most accurate mathematical account. The complete discovery cycle is therefore iterative: anonymous operations

```
→ exact semantic classes
→ recurrent identities
→ provisional basis
→ residual theory
→ adversarial countermodels
→ revised basis
→ structural representation
→ free objects and normal forms
→ finite and categorical analysis
→ equivalence audit.
```

A successful cycle ends not merely with equations but with a theory birth certificate recording how the algebra emerged, why alternatives were rejected, what has been proved, what remains computational evidence, and whether the resulting structure is

historically new, independently rediscovered, or a novel extension of known mathematics.

4. Algebraic Fertility and the Rejection of False Births

4.1 Randomness, Collapse, and Fertility

Anonymous finite operations do not automatically contain useful algebra. Most candidate tables occupy mathematically unproductive regions of operation space. At one extreme, highly irregular tables behave almost randomly. Their formal terms usually induce many distinct functions, but exact collisions among short nontrivial terms are rare. At the opposite extreme, severely collapsing tables produce large numbers of identities because most terms become constants, projections, or members of very small semantic classes. Neither extreme is a satisfactory source of algebraic theory. A random table has complexity without compressible law. A degenerate table has compressible law without meaningful complexity. Algebraic fertility lies between them.

A fertile operation system must preserve enough distinction to generate a nontrivial term universe while collapsing enough distinction to support recurrent identities. It should admit exact semantic regularities without reducing all expressions to one or two trivial behaviors. This balance may be described through the growth of term functions. Let gd(A) denote the number of distinct term functions induced by terms of complexity at most d in a finite algebra A. Very slow growth often indicates collapse. Extremely rapid growth accompanied by almost no collisions indicates near-random behavior. Productive candidates tend to display intermediate growth, in which new semantic functions continue to appear while meaningful equivalence classes also develop.

The ratio

cd(A) gd(A)/|Tdτ(X)| measures the fraction of bounded terms absorbed into already existing semantic classes. A large value of cd may signal either useful algebraic compression or trivial collapse. It must therefore be interpreted together with coordinate dependence, image size, entropy, subalgebra structure, congruence diversity, and the form of the identities creating the collisions. Fertility is not a single scalar property inherent in an operation table. It is relative to a term language, variable set, complexity bound, ecology, and selection policy. An operation may appear barren under shallow terms but reveal structure at greater depth. Another may look fertile because one short collapsing law creates thousands of derived equalities. The methodology therefore treats fertility as a vector of diagnostic measurements rather than as one unquestioned score.

A candidate is valuable when its regularities are simultaneously exact, nondegenerate, transferable, compressible, and structurally generative. These criteria define the central selection problem of Algebraic Abiogenesis.

4.2 Projection and Constant Collapse

The simplest false births arise when primitive operations ignore most or all of their inputs. For a binary operation *, the constant case has the form

```
x*y=c
```

for a fixed carrier element c. The left-projection case is

```
x*y=x,
```

and the right-projection case is

```
x*y=y.
```

Higher-arity operations admit analogous coordinate projections and constant outputs. Such operations satisfy many identities. Under a left projection,

```
(xy)z=x,
x(yz)=x,
```

and every term evaluates to its leftmost variable. A large collision catalogue follows, but it contains almost no new algebraic information. The apparent theory is simply a syntactic reflection of coordinate erasure. Exact collapse tests should therefore be applied directly to every primitive operation. For an nary operation f, define the essential-coordinate set

Ess(f)

```
{i : there exist inputs differing only at coordinate i on which f has different outputs}.
```

If Ess(f) is empty, f is constant. If Ess(f) contains only one coordinate and the operation reproduces that coordinate exactly, f is a projection. More generally, a primitive may depend on one coordinate without being a literal projection, for example by applying a unary map to that coordinate. Such operations may still be too degenerate for a search intended to discover genuinely multi-input structure.

A strict search may require

Ess(f)

```
{1,…,n}
```

for every primitive operation. A more permissive search may allow partial dependence if the signature contains several operations whose combined behavior remains structurally rich. Approximate projection collapse also matters during evolution. A table that agrees with a projection on nearly every input may dominate a fitness function based on collision count because small mutations produce many exact short identities. The search should therefore penalize high agreement with constants and projections even before exact collapse occurs.

For a binary operation, useful diagnostics include:

- the number of distinct rows;
- the number of distinct columns;
- the number of constant rows and columns;
- agreement with each coordinate projection;
- output support;
- conditional output entropy given each input coordinate.

An operation with one repeated row or column is not automatically degenerate. Repetition may reflect a meaningful quotient or congruence. The problem occurs when most operational behavior is explained by direct coordinate deletion. Projection and

constant rejection should be recorded as explicit gates rather than hidden inside an aggregate score. A candidate failing such a gate is not promoted regardless of how many identities it produces.

4.3 Output Support and Coordinate Dependence

An operation may depend formally on all coordinates while still using only a small portion of the carrier. For example, a binary operation on ten elements may output only two values. Such an operation is not constant, but its low support can cause rapid term collapse and an inflated identity count.

```
For an operation fA : An → A, define its output support by
```

Supp(fA)

```
{fA(a1,…,an) : ai∈A}.
```

The normalized support ratio is

ρ(fA) |Supp(fA)|/|A|. A candidate search can require ρ(fA) to exceed a chosen threshold. The appropriate threshold depends on the signature and research objective. Requiring full support may reject interesting retract-like structures, while allowing extremely small support encourages collapse. Entropy provides a finer measure. Let the inputs be sampled uniformly from An, and let

```
Y=fA(X1,…,Xn). The normalized output entropy is
```

Hnorm(fA)

H(Y)/log|A|. Low entropy indicates concentration on a few outputs even when the formal support is larger. High entropy indicates balanced use of the carrier. Coordinate dependence may also be measured quantitatively. For coordinate i, compare operation outputs when that coordinate varies while all others are fixed. Define

Di(fA)

as the proportion of such comparisons on which changing coordinate i changes the output. Exact essential dependence requires only Di>0, but robust dependence requires a substantially larger value. Conditional mutual information or conditional entropy can provide an alternative measurement. If changing one input rarely alters the output after the others are fixed, the operation is close to ignoring that coordinate. These measurements should not be interpreted as algebraic invariants. They are search diagnostics. Isomorphic copies of a finite algebra preserve them under carrier relabeling, but their primary function is to distinguish candidate tables likely to support nontrivial term interaction.

Multi-operation signatures require joint analysis. Two binary operations may each appear nondegenerate while being nearly identical to each other. Conversely, one operation may have limited support but provide a meaningful selector or retraction for another. The search should therefore measure both individual behavior and relational behavior among primitives. The purpose of support and dependence gates is not to impose a preferred algebraic style. It is to prevent trivial information loss from masquerading as theory formation.

4.4 Definitional Signature Collapse

A multi-operation search can appear to discover a richer theory when one primitive operation is merely a term-definable copy of another. Suppose the signature contains a unary operation u and a binary operation *. If every candidate model satisfies

```
u(x)=x*x,
```

then the unary symbol adds no independent operational content. The signature has collapsed definitionally.

The same problem occurs when two binary operations satisfy

```
x#y=x*y,
x#y=y*x,
```

or

```
x#y=t(x,y)
```

for a short term t using only the other primitives. Such identities may be mathematically legitimate, but they undermine an experiment whose purpose is to test whether multiple anonymous operations generate genuinely richer interaction. A search can incorrectly reward definitional aliases because they create large crossoperation collision families. For each primitive operation f, the system should enumerate short terms in the reduced signature

```
τ{f}
```

and test whether any term t satisfies

```
f(x1,…,xn)=t(x1,…,xn)
```

throughout the candidate ecology. If so, f is observationally definable at the tested depth. The candidate may then be:

- rejected;
- reduced to the smaller signature;
- retained as a calibration example;
- or studied explicitly as a definitional expansion.

The audit must distinguish exact definability from approximate agreement. Exact definability throughout the ecology is a strong collapse signal. Approximate definability may still indicate that evolution is approaching a redundant endpoint. Definitional collapse can also occur jointly. Two primitives may both be definable from a third, or each may be definable from the other only after introducing auxiliary terms. Detection therefore requires iterative reduction. Remove one symbol provisionally, recompute bounded term functions, and test whether the original primitive is recoverable.

A stronger analysis considers term equivalence of entire signatures. Two apparent theories may differ only because their primitive symbols are interdefinable. Such cases should not be counted as independent theory births. Definitional-collapse detection is one of the clearest examples of explanation subtraction. Once a primitive operation is explained as a term in the others, identities arising solely from that explanation should be removed from the residual theory.

4.5 Finite Coincidence

Every finite algebra satisfies identities that do not characterize a robust infinite theory. Some are consequences of its small carrier size. Others arise from accidental repetitions in the table. Still others reflect global features that disappear under minor changes of order or operation values. A short identity holding in one finite model is therefore only a local observation. Even recurrence across several models may be misleading if those models share ancestry, order, construction bias, or an unrecognized degeneracy. Cross-order validation reduces this danger but does not eliminate it. An identity may hold in several orders because the generator repeatedly converges on the same narrow structural family. The theory may still fail outside that family.

Finite coincidence is addressed through several complementary procedures. First, models should be generated under independent seeds and diverse construction mechanisms. Evolutionary lineages should not be mistaken for independent evidence. Second, carrier orders should vary. A theory that survives orders three, four, and five is less likely to be an artifact of one cardinality than a theory observed only at order three. Third, candidate laws should be tested in newly synthesized models constrained only by the provisional basis. If the larger identity family reappears, this suggests that the basis captures a generative mechanism. Fourth, adversarial searches should seek models satisfying the basis while falsifying proposed consequences. A successful countermodel distinguishes finite correlation from derivability.

Fifth, symbolic deduction should replace finite evidence whenever possible. Once an identity is derived formally from the basis, its validity no longer depends on the sampled models. The methodology must preserve a strict vocabulary:

- observed in a finite model;
- recurrent in an ecology;
- derived from a basis;
- valid in a variety;
- disproved by a countermodel.

Confusing these levels is one of the most serious risks in automated mathematical discovery. Finite coincidence is unavoidable at the beginning of the process. The objective is not to prevent all accidental identities from being recorded, but to ensure that they are progressively removed before theory promotion.

4.6 Barren Theories

A candidate theory can be exact, recurrent, and nondegenerate yet still fail to support meaningful development. Such a theory will be called barren. A barren theory may contain several displayed identities, but its apparent richness disappears after syntactic redundancy is removed. Typical sources include:

- variable renamings;
- exchange of operation symbols under an obvious symmetry;
- reversal of variables;
- transitive combinations of two previously known identities;

- direct substitutions;
- one-step contextual applications;
- repeated statements of a single collapse mechanism.

For example, suppose an ecology supports

```
s=t,
t=u,
```

and

```
s=u.
```

The third identity adds no theoretical reserve. Similarly, if a law remains unchanged under swapping x and y, its renamed copy is not an independent structural discovery. Barren theories can be detected through bounded deductive closure. Let Kd(E) be the observed bounded theory and let Σ be a provisional basis. If

Kd(E)

≈ Cnd(Σ) and Σ contains only shallow collapse laws, the apparent collision abundance has been explained without revealing deeper structure. The theory may still be mathematically legitimate. The issue is whether it is a useful endpoint for Algebraic Abiogenesis. A calibration run may intentionally recover a simple known class. A novelty-oriented run should demand more. A theory reserve may be measured by identities that:

- occur at greater syntactic depth than the basis;
- involve additional variables;
- connect distinct term architectures;
- survive independent models;
- require more than direct rewriting by one law;
- support structural representation or classification.

The search should also consider the diversity of semantic classes generated by the theory. A basis that collapses every term to one variable may explain many equations but creates little object structure. A fertile basis should produce a nontrivial quotient term algebra. Barren-theory rejection prevents the system from equating compressibility with mathematical importance. Some theories are highly compressible because they destroy almost all structure. Algebraic Abiogenesis seeks theories that compress observations while preserving enough internal complexity for further mathematics.

4.7 Criteria for a Viable Theory Embryo

A viable theory embryo is a recurrent operational regularity with sufficient evidence and internal structure to justify mathematical maturation. No single diagnostic is decisive. Promotion requires a conjunction of conditions. First, the supporting operations must pass nondegeneracy gates. Their behavior should not reduce to constants, projections, near-projections, or trivial low-support maps unless such features are part of a larger nontrivial interaction. Second, primitive operations should be distinct modulo short definability unless the experiment explicitly studies expansions or reducts.

Third, the identity fingerprint must recur across independently generated models and preferably across multiple carrier orders. Fourth, the theory should admit explanatory compression. A manageable provisional basis should account for a substantial portion of the observed congruence. Fifth, the basis should possess a consequence reserve. The observed theory should contain transferable structure beyond immediate renamings, substitutions, and one-step transitive consequences. Sixth, the theory should survive adversarial countermodel searches. Identities that fail in models satisfying the provisional basis should be removed from the claimed deductive theory.

Seventh, the basis should be examined for independence and minimality. Redundant laws should be eliminated or explicitly retained for orientation, symmetry, or interpretive clarity. Eighth, the theory should support structural maturation. At least one of the following should become accessible:

- a representation theorem;
- a normal-form system;
- a free-algebra construction;
- a finite-model formula;
- a congruence classification;
- a nontrivial word problem;
- a subdirect decomposition;
- a natural categorical or combinatorial interpretation.

Ninth, the theory should undergo an equivalence audit. A candidate cannot be described as new merely because its displayed equations have not been recognized. Duality, term equivalence, opposite operations, definitional extension, reducts, and alternative bases must be considered. Tenth, the entire discovery path should be reproducible. The operation generators, term bounds, fitness components, model ecologies, basis choices, rejected identities, countermodels, and software versions must be recorded.

These requirements turn theory promotion into a falsifiable process. A candidate may fail at any stage. Most should fail. A methodology that promotes every recurrent collision family is not discovering algebraic organisms; it is accumulating artifacts. A viable theory embryo is therefore not defined only by what it satisfies. It is defined by the tests it has survived and by the mathematical development it can sustain. The rejection architecture is central to Algebraic Abiogenesis. In biological abiogenesis, most molecular arrangements do not become living systems. In algebraic abiogenesis, most anonymous operations do not become viable theories. The purpose of fertility analysis is to distinguish transient formal regularities from structures capable of becoming abstract mathematics.

Chapter 11: Countermodels as Selective Pressure

Adapted from Mathematical It from Bit (2026), Sections 5-7.

5. Explanation-Subtracted Fertility and Recursive Theory Peeling

5.1 The Failure of Raw Collision Counts

The earliest versions of Algebraic Abiogenesis treated the number of exact term-function collisions as a principal signal of mathematical fertility. This was a reasonable starting hypothesis. If many formally distinct expressions induce the same functions throughout a finite algebra, the operation table evidently contains compressible regularity. A table producing no collisions among thousands of short terms offers little immediate material from which to infer an equational theory. Collision abundance, however, is not equivalent to theoretical depth. One short identity can generate a very large family of derivative equalities. If every instance, substitution, variable renaming, and contextual consequence is counted separately, a single elementary mechanism may appear to constitute a rich theory.

Consider a ternary operation m satisfying

```
m(x,x,x)=x.
```

This law is nontrivial and may occur in nondegenerate finite algebras whose operation depends on all three coordinates. Yet the law immediately generates many identities. For arbitrary terms s,

```
m(s,s,s)=s.
```

If s itself contains nested occurrences of m, then each substitution produces additional equalities at greater depth. Applications inside larger contexts multiply the number further. A

bounded collision catalogue may consequently contain dozens or hundreds of equations, even though all of them express one diagonal contraction. Raw collision count therefore measures the size of the observed congruence relation within the chosen term universe, but it does not measure how many distinct explanatory mechanisms are present. The distinction is analogous to the difference between data volume and model complexity. A simple rule may explain a vast dataset. Counting every datum as an independent discovery exaggerates the amount of structure that remains unexplained. This failure became decisive in the ternary calibration. The initial cross-order ecology produced 41 recurring exact identities among 300 generated terms. The models were not constants or projections, and their ternary operations retained meaningful coordinate dependence. Under a raw-collision criterion, the result appeared to be a successful theory embryo.

Once the diagonal law

```
m(x,x,x)=x
```

was oriented as the rewrite rule

```
m(u,u,u)→u,
```

every one of the 41 recurring identities was explained. No residual law remained. The apparent multi-identity theory was therefore a one-law theory whose consequences had been counted repeatedly. A more serious version of the same problem occurs when a shallow identity produces consequences that look syntactically diverse. Two terms may differ substantially in depth, variable distribution, or tree shape while still being related by one local contraction. Visual complexity does not guarantee explanatory independence. A raw-collision objective also distorts evolutionary selection. Candidate operations satisfying a single highly generative collapse law can dominate candidates whose theories contain several subtler mechanisms but fewer immediate collisions. Evolution then optimizes for semantic compression without distinguishing informative compression from destructive collapse.

The remedy is to make fertility conditional on what has already been explained. The system must not continue rewarding identities merely because they are true. It must ask whether they contain information not generated by the current provisional theory. This leads to explanation-subtracted fertility.

5.2 Bounded Observed Theory

Let τ be a finitary signature, X a finite variable set, and Tdτ(X) a bounded universe of formal terms. The bound d may refer to depth, size, operation count, or another fixed complexity measure. For an ecology E of finite τ-algebras, define the bounded observed theory

Kd(E)

```
{(s,t) in Td_tau(X)^2 : A |= s=t for every A in E}.
```

Equivalently,

Kd(E)

ΘE∩(Tdτ(X)×Tdτ(X)), where ΘE is the observational congruence induced by the ecology. The relation Kd(E) is exact on the declared finite universe. Every pair in it has been evaluated under every assignment in every model in E. It is not a sample estimate. Because equality is symmetric and transitive, Kd(E) can be represented as a partition of Tdτ(X) into semantic classes. If one class contains terms t1,t2,…,tk, then all k(k-1)/2 unordered equalities among them belong to the observed theory. This creates an immediate counting ambiguity. Should such a class count as one discovered mechanism, k-1 independent equalities relative to a representative, or k(k-1)/2 pairwise collisions? None of these counts alone identifies how much deductive structure is present.

The partition representation is therefore more informative than a flat equation list. It reveals whether the observed theory consists of many small equivalence classes or a few enormous classes generated by strong collapse. For computation, one shortest or otherwise preferred term may be selected as the representative of each class. Every other member is stored as an equality to that representative. This produces a minimal spanning description of the finite equivalence class, but it still does not determine which

equalities are consequences of others under substitution and context. The observed theory depends on the ecology. If E is enlarged by adding countermodels, Kd(E) can only shrink. It also depends on the term bound. Increasing d can reveal new identities and

new consequences of old identities. Therefore, Kd(E) is not presented as a final theory. It is the exact observational target that a provisional basis should explain at one stage of development. The system records several properties of Kd(E):

- the number of terms;
- the number of semantic classes;
- the distribution of class sizes;
- the number of variable-essential classes;
- the complexity range within each class;
- recurrence across carrier orders;
- the proportion involving each primitive operation;
- the extent to which identities connect different term architectures.

These measurements help distinguish shallow collapse from richer organization. A class containing a variable and thousands of increasingly nested terms may reflect one erasing law. Several moderate classes linking distinct operations may indicate multiple interacting mechanisms. The bounded observed theory is thus the empirical object of explanation. It contains everything the current ecology says is indistinguishable within the current syntactic window.

5.3 Bounded Deductive Closure

Let Σ be a provisional set of identities. The complete equational consequence relation of Σ is generally infinite. To compare it with Kd(E), the method computes a bounded deductive closure Cnd(Σ) inside Tdτ(X).

```
A pair (s,t) belongs to Cnd(Σ) when s=t can be obtained within the bounded environment by
```

sound equational operations such as:

- reflexivity;
- symmetry;
- transitivity;
- substitution of bounded terms for variables;
- replacement inside permitted term contexts;
- application of oriented rewrite rules;
- normalization under a certified convergent system;
- bounded congruence saturation.

The precise computational procedure may vary, but every inferred equality must be a sound consequence of Σ. Incompleteness is acceptable during exploratory compression; unsoundness is not. When a convergent rewrite system R for Σ is known, bounded closure is straightforward: (s,t)∈Cnd(Σ) exactly when

```
NFR(s)=NFR(t).
```

Before convergence has been established, the system may use finite congruence saturation. Start with the declared instances of Σ in the bounded term universe, close under symmetry and transitivity, and repeatedly impose compatibility under every operation context that remains within the bound. For example, if s≡t has been established and f is binary, then bounded closure includes f(s,u)≡f(t,u) and f(u,s)≡f(u,t) for every bounded term u for which the resulting terms remain in Tdτ(X). Substitution is equally important. If the basis contains

```
m(x,y,y)=x,
```

then replacing x by s and y by t gives

```
m(s,t,t)=s.
```

A collision catalogue that counts each such instance independently will overstate the explanatory content unless substitutions are included in Cnd(Σ). Bounded closure is sensitive to the chosen universe. A derivation between two small terms may pass through an intermediate term larger than d. A closure procedure restricted to Tdτ(X) could

miss such a proof. For this reason, the implementation may use a larger derivation universe than the observation universe, or employ normalization that avoids large intermediate expansions. The relation Cnd(Σ) is used as an explanatory model of Kd(E). Ideally,

```
Cnd(Σ)=Kd(E).
```

When

Cnd(Σ)⊂Kd(E), some observed identities remain unexplained.

When

Cnd(Σ) contains equalities not observed in E, one of two situations holds. Either the ecology is too small to exhibit all genuine consequences of Σ, or Σ is too strong and predicts identities not supported by the models. Since every model in E should satisfy the provisional basis, sound consequences of Σ must also hold in E. A discrepancy therefore indicates an implementation error, an incorrect basis certification, or an ecology containing models that do not actually satisfy Σ. This containment audit is a fundamental consistency check: Cnd(Σ)⊆Kd(E) whenever every algebra in E satisfies Σ.

5.4 Residual Theory

Define the bounded residual theory by

$R^d(E|\Sigma) = K^d(E) \setminus Cn^d(\Sigma)$. The residual contains the observed identities that are not yet explained by the provisional basis. This set is the principal object of further search. Once an identity has been accounted for deductively, it should cease contributing to the fertility score. The search then concentrates on whatever structure remains. Suppose a candidate ecology supports 500 recurring identities. If 490 follow from one short law, the relevant question is not whether the theory has 500 identities. It is whether the remaining ten form a recurrent, nondegenerate, compressible pattern.

The residual may be empty. This does not mean the experiment failed. It means the current basis completely explains the bounded observed theory. The resulting theory may

be retained as a calibration, rejected as too shallow, or promoted if the basis itself produces mathematically significant structure. The residual may also be fragmented. Different models may share the provisional basis but support different additional laws. In this case, the common ecology may have little residual theory even though subpopulations contain strong theory embryos. Theory-ecology clustering is then required before further peeling. Residual identities can be ranked by:

- syntactic simplicity;
- recurrence frequency;
- carrier-order diversity;
- semantic nondegeneracy;
- number of unexplained classes merged;
- proof distance from the current basis;
- capacity to explain other residual identities;
- resistance to adversarial countermodels.

The residual should also be quotient-aware. If one semantic class contains several unexplained terms, the system should avoid treating all pairwise equalities as separate candidates. It may select a representative law that merges the largest unexplained components.

The difference operation in

$K^{d}(E) \setminus Cn^{d}(\Sigma)$ is conceptually simple but methodologically transformative. It changes the optimization target from total regularity to unexplained regularity. The system is no longer rewarded for rediscovering the consequences of what it already knows.

5.5 Recursive Peeling

Recursive theory peeling repeatedly extracts explanatory laws from the residual theory.

Begin with

```
Σ0=ø.
```

The initial residual is

R0

Kd(E|Σ0)

Kd(E)

apart from reflexive equalities and syntactic identities. Select a candidate law σ1 from R0. The preferred candidate is usually short, transferable, nondegenerate, and capable of explaining a large portion of the residual.

Set

```
Σ1={σ1}
```

and compute

R1

Kd(E|Σ1). If R1 is nonempty, choose another law σ2 that best compresses R1. Continue: Σi+1

```
Σiυ{σi+1},
```

Ri+1 Kd(E|Σi+1). The process stops when one of several conditions is met:

- the residual is empty;
- the residual lacks cross-order recurrence;
- all remaining identities are degenerate;
- no candidate explains a meaningful portion of the residual;
- adversarial countermodels destroy the remaining pattern;
- the computational budget is exhausted.

Peeling differs from ordinary greedy basis extraction because the residual is recomputed semantically after every explanation. A candidate law may make another candidate redundant.

Conversely, subtracting a dominant law may reveal a smaller pattern that was previously hidden inside a large equivalence class. The order of extraction matters. Choosing a long law before a short generative law can produce an unnecessarily large basis. Therefore the system generally favors the shortest candidate among those with comparable explanatory power. A minimum-description objective can formalize this tradeoff. Let L(Σ) measure the complexity of the basis and L(R) the remaining unexplained information. The system seeks to minimize L(Σ)+λL(R), where λ determines the penalty for leaving observations unexplained. The process resembles scientific explanation. A provisional law is accepted not merely because it fits observations but because it renders many observations unsurprising. The remaining anomalies become the subject of the next explanatory cycle.

Recursive peeling can also identify layered theories. The first law may govern one primitive operation, the second may connect two operations, and the third may impose a higher-order compatibility condition. The final basis then reflects a hierarchy of mechanisms rather than an unstructured list of equations. Every peeling step should be recorded in the theory birth certificate:

- the residual before selection;
- the selected identity;
- its explanatory gain;
- the residual after selection;
- rejected alternatives;
- countermodels;
- the stopping reason.

This history makes theory formation auditable. It shows not only which basis was chosen but why.

5.6 Consequence Reserve

A consequence reserve is the body of stable deductive structure generated beyond the immediate appearance of the basis.

A theory with no reserve may consist only of its displayed laws, variable renamings, direct substitutions, and trivial contextual applications. Such a theory can be exact but mathematically barren. A theory with a substantial reserve produces identities that:

- occur at greater depth than the basis;
- connect distinct term shapes;
- involve several variables or operations;
- recur in independently generated models;
- require multi-step deduction;
- imply structural restrictions on models;
- support canonical representations or normal forms.

Let Σ be a provisional basis. One bounded reserve measure is Resd(Σ,E) |Cnd(Σ)∩Kd(E)|-B(Σ), where B(Σ) discounts reflexive identities, the basis itself, direct variable renamings, and other declared trivial consequences. A normalized version may divide by the number of possible bounded term pairs or by basis description length. Reserve should not be measured only numerically. One collapse law may generate an enormous number of substitutions while contributing little structural diversity. The reserve must therefore be stratified by consequence type. Useful reserve dimensions include:

- depth extension;
- variable extension;
- operation interaction;
- architectural diversity;
- proof length;
- semantic class diversity;
- model-theoretic consequences.

A law such as

```
m(x,y,y)=x
```

has a large numerical reserve because it reduces every instance m(s,t,t). Yet its mechanism remains simple. It is useful as a calibration because it demonstrates the method, but it may not satisfy a novelty-oriented fertility threshold. By contrast, a small basis that generates several qualitatively different reduction patterns, model decompositions, and finite invariants exhibits a stronger reserve. The consequence-reserve criterion is not intended to rank all mathematics universally. Some important theories have simple axioms and simple consequences. It is a search policy for identifying candidates likely to sustain extended development. A candidate theory may be promoted even with a modest reserve if it has another compelling feature, such as an

unexpected representation theorem or a novel interaction between operations. The reserve is evidence, not an absolute definition of value.

5.7 Information-Theoretic Interpretation

Explanation-subtracted fertility admits an information-theoretic interpretation. The bounded observed theory Kd(E) contains finite relational information about which terms are identified by the ecology. A basis Σ acts as a compressed generative description of part of that information. The deductive closure Cnd(Σ) is the portion reconstructed from the compressed description. The residual Rd(E|Σ) is the information not yet reconstructed. Theory formation can therefore be viewed as successive coding: observed equivalence data

```
→ compact identity basis
→ deductive reconstruction
→ residual data
→ additional basis law.
```

A successful basis has a high compression ratio: a short description generates a large, accurate body of observed equivalence.

However, maximal compression is not automatically desirable. The identity

```
x=y
```

would collapse the entire term algebra in every nonempty context if admitted, producing extreme compression but destroying nearly all structure. The method therefore constrains compression by nondegeneracy and model viability. The relevant objective is structured compression: remove redundant distinctions while preserving a rich quotient algebra.

The quotient

Tτ(X)/ΘE can itself be interpreted as a compressed semantic representation. Terms in the same class are different encodings of one observational object. A basis describes why those encodings are equivalent. Countermodels add information by restoring distinctions. If a proposed identity fails in a newly generated model, the observational congruence becomes finer. The description of the theory must then become more discriminating. This yields an alternating process: compression by explanation, refinement by counterexample. The first reduces apparent complexity. The second prevents overcompression. Algebraic Abiogenesis can therefore be understood as a search for an optimal identity policy between two failures:

- undercompression, in which every formal term remains a separate object;
- overcompression, in which degenerate laws erase nearly all distinctions.

A fertile theory occupies the intermediate regime. It forms stable objects by identifying many expressions, but it preserves enough difference to support nontrivial operations, free constructions, and model diversity. This is the precise methodological content of Mathematical It from Bit in the present setting. Finite operational data determine provisional distinctions. Recurrent equivalences erase some distinctions. Short laws compress the erasures. Countermodels restore distinctions erased without sufficient justification. The surviving quotient becomes the emergent algebraic object.

Explanation subtraction is consequently not only a technical improvement to a fitness function. It is the mechanism by which the discovery process distinguishes repetition from information, consequences from causes, and algebraic appearance from algebraic birth.

6. Theory Ecologies and Cross-Order Formation

6.1 Models as Theory-Bearing Organisms

A finite algebra in Algebraic Abiogenesis is not treated merely as a table of outputs. It is treated as a bearer of a bounded equational phenotype. Its primitive operation tables determine how formal terms behave, and those term behaviors determine an observational theory. For a finite τ-algebra A and bounded term universe $T^d\tau(X)$, define its bounded identity fingerprint by K^d(A) =

$$\{(s,t) \in [T\tau^d(X)]^2 : A \vDash s=t\}.$$

Equivalently, $K^d(A)$ records the partition of the bounded term universe into classes of terms inducing the same function on A. Two finite algebras may have operation tables that look entirely unrelated while supporting nearly identical identity fingerprints. Conversely, two tables differing in only a few entries may generate sharply different term theories. The relevant similarity for theory formation is therefore semantic rather than tabular. This distinction is fundamental. Algebraic Abiogenesis is not attempting to cluster finite operations by visual or numerical resemblance. It seeks populations whose members express the same transferable algebraic regularities.

The biological analogy is limited but useful. A finite algebra is like an organism whose operation table constitutes its finite substrate and whose recurrent term identities constitute its theoretical phenotype. Mutation changes the substrate. Selection acts partly on the phenotype. Different substrates may express the same theory, just as different physical realizations may instantiate the same abstract law. The theory fingerprint should usually be explanation-subtracted. If Σ is the current provisional basis, define the residual fingerprint $R^d(A \mid \Sigma)$

$$R^d(A|\Sigma) = K^d(A) \setminus Cn^d(\Sigma).$$

This removes identities already explained by the current theory and represents only the algebra's remaining unexplained behavior. Two models that both satisfy a shallow law may share thousands of raw identities. After subtraction, they may have no unexplained theory in common. Such models belong to the same broad equational environment but do not necessarily support the same next theory layer. Residual fingerprints allow the ecology to evolve recursively. At the beginning, models are compared by their total bounded theories. After one law is extracted, they are compared by what remains. A population that initially appears homogeneous may divide into several distinct theoretical lineages once its dominant common law is removed.

The model record should therefore contain at least four levels of information: the primitive operation tables, the complete term-function fingerprints, the identities explained by the current basis, and the residual identities not yet explained. This layered representation prevents the method from confusing shared ancestry with shared theory and shared shallow laws with shared deeper structure.

6.2 Similarity and Overlap

To construct a theory ecology, the method requires a measure of similarity between finite algebras. The simplest measure compares their residual identity sets.

For two models A and B, define the intersection score

$I^{d}\Sigma(A,B)$ $|R^{d}(A|\Sigma) \cap R^{d}(B|\Sigma)|$.

A normalized Jaccard score is

$J^{d}\Sigma(A,B)$ $|R^{d}(A|\Sigma) \cap R^{d}(B|\Sigma)| / |R^{d}(A|\Sigma) \cup R^{d}(B|\Sigma)|$.

The intersection score emphasizes the absolute amount of common unexplained theory. The Jaccard score emphasizes the proportion of the two theories that overlaps. Both are useful, but they answer different questions. A model with a large residual identity set may share many identities with another model while also possessing many additional laws. Its absolute overlap may be large even when the normalized similarity is moderate. Such a pair may represent one theory plus a specialized subtheory. A high Jaccard score may instead arise from two models having very small residual theories. If each supports the same three unexplained identities, their normalized similarity is perfect, but the common theory may be too small to justify promotion.

Accordingly, theory-ecology construction should use several simultaneous thresholds: a minimum absolute overlap, a minimum normalized overlap, a minimum number of nontrivial semantic classes, a minimum carrier-order diversity, and a minimum residual complexity. Identity pairs should also be weighted. Equalities involving constants, projections, or singlevariable collapses may receive lower weight. Identities connecting terms with distinct tree architectures, multiple variables, or several operation symbols may receive greater weight. Let w(s,t) be an identity weight. Then the weighted overlap is $W^{d}\Sigma(A,B)$ $\sum$ w(s,t), summed over all identities common to both residual fingerprints.

The weight function may include: the number of essential variables, the difference in term architectures, the number of primitive operations represented, the syntactic distance between the two terms,

the minimum proof depth under the current basis, and the rarity of the identity across the full population. Rare but recurrent identities may be especially informative. An equation appearing in nearly every generated table may express a generation bias rather than a selective theory. An equation shared by a small, cross-order cluster may identify a distinct algebraic lineage. Similarity should also be computed modulo obvious symmetries. Variable renaming, exchange of equally typed operation symbols, reversal of binary arguments, and carrier relabeling can make equivalent theories appear different at the syntactic level. A canonical identity representation can reduce this duplication. Variables are renamed in order of first appearance, symmetric operation transformations are normalized when appropriate, and the shorter side of an unoriented equality is placed first under a fixed ordering.

Care is required because not every apparent symmetry is mathematically legitimate. Two operation symbols should not be exchanged merely because they have the same arity. Such an exchange is valid only when the research question explicitly treats the

signature as anonymous under symbol permutation or when a term equivalence has been established. The resulting similarity matrix does not itself define a theory. It supplies evidence about which models should be examined together.

6.3 Theory-Ecology Graphs

A theory ecology can be represented as a graph

$G^{d}\Sigma$ (V,E), where each vertex represents a finite algebra and an edge connects two vertices when their residual-theory similarity exceeds a selected threshold. The graph may be unweighted, but a weighted graph is more informative. Edge weight records the degree of common unexplained theory. Vertex metadata records carrier order, generation lineage, primitive-operation diagnostics, and basis satisfaction. Several graph structures are useful. A connected component represents a family of models linked by overlapping residual theories. It may contain more than one theory embryo if the overlap is chained rather than globally shared.

A clique represents a stronger form of coherence. Every pair of models in a clique shares substantial residual structure. A cross-order clique containing independently generated models is a promising theory-bearing ecology. A dense subgraph or community may be useful when strict clique requirements are too severe. Real model ecologies may contain noisy members, specializations, and partial theories. Community-detection methods can identify groups with high internal theoretical overlap and low external overlap. A core-periphery structure may reveal a general theory together with specialized extensions. Core models share a common basis, while peripheral models satisfy additional identities.

Graph construction should occur after major shallow explanations have been subtracted. Otherwise one dominant identity may connect almost every model, producing one large but uninformative component. The ternary calibration illustrates the importance of this step. Before subtraction, several nondegenerate models appeared related because they all satisfied diagonal idempotence. After removing every consequence of that law, most of the apparent commonality disappeared. A smaller mixed-order clique remained with a substantial residual theory. The graph can be recomputed after each peeling stage: $G^{d}\Sigma_{0}$, $G^{d}\Sigma_{1}$, $G^{d}\Sigma_{2}$, … As Σ grows, edges supported only by already explained identities vanish. Persistent clusters indicate deeper common structure.

This evolving graph records the theoretical differentiation of the model population. Early stages may show broad shared collapse. Later stages expose separate algebraic lineages. Graph edges should not be interpreted as proof that two models belong to the same variety. They indicate bounded observational affinity. Formal theory formation occurs only after a common basis has been extracted and verified. The ecology graph is therefore a discovery instrument. It identifies where to search for a transferable basis.

6.4 Cross-Order Cliques

A cluster composed entirely of models of one carrier size may reflect cardinality-specific behavior. Algebraic Abiogenesis therefore gives special priority to cross-order clusters. Let C be a clique or dense community in the theory-ecology graph. Define its order spectrum by

Ord(C)

```
{|A| : A∈C}.
```

A cross-order theory embryo should satisfy

|Ord(C)| ≥ 2, with stronger evidence supplied by three or more distinct carrier sizes. Carrier-order diversity matters because finite algebras can satisfy identities for accidental cardinal reasons. An operation on a three-element set may collapse terms because every sufficiently long trajectory enters a short cycle. That mechanism may disappear at order four. Recurrence across orders suggests that the common identities arise from a structural feature independent of the exact number of elements. Order diversity alone is not enough. Models at different orders may be obtained by trivial padding, direct products, duplicated fibers, or homomorphic inflation of one seed. Such models are not fully independent.

The birth certificate should record whether models were: generated independently, obtained by mutation, constructed as products, formed as quotients, embedded as subalgebras, or synthesized subject to a provisional basis. A cross-order clique is strongest when its members arise through distinct search trajectories and are not linked by obvious structural constructions.

Given a candidate cluster C, define its common bounded theory

$K^d(C) = \bigcap \{ K^d(A) : A \in C \}$. Relative to a provisional basis Σ, define its common residual

$$R^d(C|\Sigma) = K^d(C) \setminus Cn^d(\Sigma).$$

This common residual is the raw material for the next basis-extraction stage. Cluster size and theory size must be balanced. A large cluster may share only shallow laws. A small cluster may share a rich but narrow theory. Promotion thresholds should therefore consider both population support and theoretical content.

The system may assign a cross-order support score such as

S(C)

|C| · |Ord(C)| · log(1+|R^d(C|Σ)|), modified by penalties for shared ancestry and degeneracy. This score is heuristic. Its role is to prioritize clusters for deeper analysis, not to certify mathematical importance. Cross-order cliques also support theory transfer tests. Once a basis is extracted from one clique, new models of carrier orders not represented in the original cluster can be synthesized under the basis. If their residual fingerprints match the original theory, the candidate gains evidence of structural portability.

6.5 Ecological Stability

A provisional theory should remain recognizable as the ecology changes. This property will be called ecological stability. Let E be a supporting ecology and Σ a proposed basis. Generate an enlarged ecology E′ E∪F, where F contains new independently produced models satisfying Σ.

The theory is stable at bound d when the new models preserve most of the predicted identity structure: $K^d(E') \approx$

$$Cn^d(\Sigma),$$

and when any remaining residual continues to form coherent cross-order clusters. A stronger test adds adversarial models specifically optimized to falsify high-value candidate consequences. If those consequences survive, confidence in the basis increases. If they fail, the ecology refines and the theory must be revised. Several stability measurements are useful. Identity retention measures the proportion of previously common identities remaining common after enlargement. Cluster retention measures whether the original theory-bearing models remain connected under residual similarity. Basis retention measures whether the same compact basis is selected after the ecology is expanded.

Residual retention measures whether the same unexplained identity family persists. Order transfer measures whether the theory appears in carrier sizes absent from the original discovery population. Seed stability measures whether independent runs recover equivalent bases or equivalent theory clusters. A theory that disappears when one additional model is introduced was likely a finite coincidence. A theory that persists under repeated enlargement is a stronger candidate. Ecological stability is related to but distinct from formal validity. If every new model is constrained to satisfy Σ, then all formal consequences of Σ must survive. Residual identities, however, may disappear. Their disappearance reveals that they were properties of the original ecology rather than consequences of the basis.

This distinction allows the method to separate the core theory from specialized finite subclasses. Suppose E satisfies Σ and also an additional identity ρ. A newly synthesized model satisfies Σ but falsifies ρ. Then ρ is removed from the general theory but may still define a subvariety or

subecology. The model population has differentiated into a general lineage and a specialized branch. Theory ecologies can therefore support not only one theory birth but the beginning of a theory taxonomy. Clusters may correspond to varieties, subvarieties, expansions, reducts, or finite exceptions. A mature methodology should preserve these branches rather than discarding every rejected identity. A law rejected as a consequence of the general basis may still define an interesting specialization. Ecological refinement thus creates a hierarchy: common foundational laws, cluster-specific extensions, model-specific accidents. This hierarchy is often closer to actual algebraic organization than a single flat identity list.

6.6 Theory Birth Certificates

Every promoted theory should be accompanied by a theory birth certificate. The certificate records enough information for another investigator to reproduce, audit, challenge, and reinterpret the result. The certificate begins with the signature: the operation symbols, their arities, whether symbol permutations were treated as symmetries, and the variable set used in discovery. It records the model ecology: carrier orders, number of models, complete operation tables or generation hashes, random seeds, lineages,

fitness conditions, and whether each model was random, evolved, constrained, or adversarial. It records the observational window: term-generation grammar, complexity bound, number of generated terms, assignment domains, evaluation method, and exact collision counts. It records the rejection diagnostics: coordinate dependence, output support, entropy, projection agreement, constant behavior, definitional-collapse tests, and primitive-operation distinctness. It records the ecology analysis: identity fingerprints, residual fingerprints, similarity metrics, graph thresholds, cluster memberships, cross-order support, and shared ancestry corrections.

It records the peeling history:

the initial observed theory, the first selected explanation, the deductive closure generated, the residual after subtraction, each subsequent basis law, and the stopping condition. It records countermodels: which candidate identities were challenged, the operation tables of separating algebras, the assignments witnessing failure, and whether the countermodels were incorporated into the ecology. It records the final mathematical status: finite observation, cross-order recurrence, provisional basis, formally derived consequence, proved variety theorem, known rediscovery, candidate-original extension, or unresolved conjecture. It records structural maturation: rewrite systems, normal forms, free-algebra constructions, finite-model counts, congruence data,

representation theorems, countermodel compilers, and formal proof certificates. It records the equivalence audit: alternative operation names, opposite and dual structures, known varieties considered, definitional extensions, term equivalences, reducts, expansions, and literature-search results. Finally, it records reproducibility information: software version, dependency versions, source-code hashes, data-file hashes, execution commands, hardware assumptions, and smoke-test results. The birth certificate should distinguish machine-generated outputs from human or languagemodel mathematical interpretation. It should state which identities were discovered automatically, which basis was selected algorithmically, which proofs were synthesized, and which structural ideas were introduced through guided reasoning.

This distinction is essential for evaluating claims of autonomy.

The certificate also preserves failed paths. Rejected candidate theories, barren clusters, collapsed signatures, and rediscovered structures should remain part of the record. Negative runs reveal how the method distinguishes genuine theory formation from false birth. A theory birth certificate is therefore more than supplementary documentation. It is the unit of scientific accountability for Algebraic Abiogenesis.

The central claim of a run is not merely

"these equations were found."

It is

"these anonymous operations generated this observational congruence; this ecology supported this transferable residual; this basis explained it; these countermodels removed alternatives; these structural results followed; and this equivalence audit determined its mathematical status." Only with that complete chain can an apparent algebraic birth be evaluated independently. Theory ecologies convert isolated finite algebras into populations of competing mathematical possibilities. Cross-order recurrence identifies regularities that survive changes of scale. Explanation subtraction reveals hidden lineages. Countermodels refine the population. Birth certificates preserve the complete developmental history.

This ecological level is where Algebraic Abiogenesis ceases to be equation mining and becomes theory formation.

7. Counterexample-Guided Theory Formation

In conventional mathematics, a counterexample is often treated as the terminal event in the life of a conjecture. A statement is proposed, a violating structure is found, and the statement is rejected or weakened. In Algebraic Abiogenesis, countermodels play a more active role. They do not merely terminate false theories; they reshape the model ecology from which the next theory is inferred. Let E be a finite ecology and let Σ be a provisional basis extracted from its common observed theory. Suppose an additional identity

```
ρ : s=t
```

holds in every algebra currently contained in E. This finite recurrence does not establish that ρ follows from Σ. To determine whether ρ belongs to the general theory, the system searches for a finite algebra B such that $B \models \Sigma$ but $B \not\models \rho$. If such a B is found, it is not discarded after falsifying ρ. It is added to the ecology:

```
E′=Eu{B}.
```

The observational congruence then refines: $\Theta E' \subseteq \Theta E$. Terms previously identified only because every old model agreed may become distinguishable. The residual theory is recomputed, candidate clusters may divide, and the provisional basis may be revised. Countermodels therefore function as selective pressure. They eliminate equations that do not survive outside the original finite sample and favor bases whose consequences remain stable under deliberate attack. This process differs from passive validation. Randomly generating more models may eventually expose false consequences, but adversarial synthesis focuses computational effort on the weakest points of the current theory. The system asks not only which models naturally arise, but which models would most efficiently distinguish competing explanations.

A provisional theory may be challenged at several levels. Individual candidate consequences can be targeted. Entire basis laws can be removed one at a time to test independence. Structural claims such as associativity, commutativity, local finiteness, or cancellation can be attacked. Proposed equivalences with known varieties can be tested by searching for models lying in one class but not the other.

The counterexample cycle may be written

candidate basis

```
→ predicted consequences
→ targeted model synthesis
→ separating algebra
→ ecological refinement
→ revised basis.
```

This loop continues until the remaining claims are formally derived, resistant to the current search, or explicitly classified as unresolved. Counterexample-guided theory formation changes the epistemic status of the model ecology. Its members are no longer only naturally selected examples. Some are deliberately constructed boundary cases. The resulting ecology becomes a record of both positive expression and negative discrimination. A robust theory is not one that fits only the models from which it was induced. It is one that survives models designed to make it fail.

7.2 Adversarial Model Synthesis

Adversarial synthesis begins with a provisional basis Σ and a target identity ρ. The objective is to construct a finite τ-algebra satisfying every law in Σ while violating ρ. For a fixed carrier size n, the synthesis problem can be expressed as a finite constraint system. Every entry of every primitive operation table is an unknown carrier value. Each identity in Σ contributes constraints over all assignments of its variables. The negation of ρ requires at least one assignment on which its two sides evaluate differently.

For example, if

```
ρ : s(x,y)=t(x,y),
```

then falsification requires elements a,b such that sA(a,b)≠tA(a,b). The search may be performed by exhaustive enumeration at very small orders, by satisfiability solving, constraint programming, stochastic mutation, evolutionary optimization, or hybrid methods. Exhaustive search provides the strongest finite certificate. If every operation table of order n is tested, the absence of a countermodel establishes that none exists at that order. The cost, however, grows rapidly. A single binary operation on n labeled elements has $n^{(n^2)}$ possible tables. Two binary operations have $n^{(2n^2)}$ possible interpretations. Constraint-based synthesis avoids enumerating entire tables. The defining identities restrict only certain relationships among entries. These restrictions can be propagated before all values are assigned. Partial tables that already violate Σ are abandoned.

Evolutionary countermodel search uses a fitness objective favoring exact satisfaction of Σ and disagreement between the two sides of ρ. Near-satisfaction may guide the search, but the final countermodel must be certified exhaustively.

A useful lexicographic objective is: first, minimize the number of violated basis instances; second, maximize the number of assignments falsifying ρ; third, maximize primitive-operation nondegeneracy; fourth, minimize carrier order and table complexity. The first priority ensures that a candidate cannot compensate for violating Σ by strongly falsifying ρ. Only exact models of the basis are accepted. Countermodel synthesis should

begin at the smallest possible order. A minimal finite countermodel is often mathematically informative. It may reveal the exact structural distinction missing from the proposed proof. It also gives the simplest reproducible certificate.

The search should not assume that a target law fails. Failure to find a countermodel may indicate that the law is derivable, that the required countermodel is larger than the current bound, or that the search procedure is insufficient. The result must therefore be reported accurately: countermodel found; no countermodel exists through order n by exhaustive search; no countermodel was found within a bounded heuristic search; or identity formally derived from Σ. These outcomes have different evidential strengths. Adversarial synthesis can also target model properties rather than equations. To test whether Σ implies associativity, search for a model of Σ and elements x,y,z satisfying $(xy)z \neq x(yz)$.

To test whether all models are commutative, seek x,y with

$xy \neq yx$. To test local finiteness, construct increasingly large finitely generated models or demonstrate unbounded free-algebra growth. The adversarial system therefore acts as a mathematical experimentalist. It constructs operational worlds in which a proposed regularity would be forced to reveal whether it is law or coincidence.

7.3 Independence Testing

A finite basis should be tested for redundancy. Let

$$\Sigma=\{\sigma_1,\ldots,\sigma_k\}.$$

For each i, define

$$\Sigma_i=\Sigma\{\sigma_i\}.$$

The identity σ_i is independent of the remaining basis when there exists an algebra A_i satisfying $A_i \models \Sigma_i$ but $A_i \not\models \sigma_i$. Such an algebra is an independence countermodel.

The complete collection

$$\{A_1,\ldots,A_k\}$$

constitutes a finite independence certificate for Σ. It proves that no displayed identity can be deleted without enlarging the model class. Independence is distinct from minimality in several senses. A basis may be irredundant but equivalent to another basis containing fewer equations. Two laws may be independent of each other while their conjunction is equivalent to a single more complex identity. Algebraic Abiogenesis therefore distinguishes: irredundancy of the displayed basis; minimum number of basis identities; minimum total syntactic complexity; and minimum complexity under alternative signatures or definitional extensions. The earliest binary calibration exposed this distinction. A two-law basis was correctly shown to be irredundant because neither law followed from the other. Later analysis revealed that the same variety could be defined by one different identity. The two-law basis was irredundant but not cardinality-minimal.

The basis extractor should therefore perform several audits. First, remove each law and search for an independence countermodel.

Second, search the observed identity catalogue for a single equation or smaller subset whose bounded closure explains the same theory. Third, test whether combinations of laws can be compressed into one identity. Fourth, compare alternative orientations for rewriting and structural clarity. A nonminimal basis may still be preferred for exposition. Separate left and right laws may reveal symmetry more clearly than one compressed equation. A larger basis may orient into a simpler convergent rewrite system. The theory birth certificate should state why a particular presentation is retained. Finite independence countermodels are especially valuable because they transform an abstract nonderivability claim into explicit data. The carrier, operation tables, and falsifying assignment can be checked independently without reproducing the original search.

When no finite countermodel is found, symbolic deduction becomes necessary. A law may follow from the others only through substitutions or contexts beyond the finite search bound. Automated equational reasoning, completion, and model finders should therefore be used together. Independence testing also reveals the functional roles of the basis laws. Removing one identity may permit projection collapse. Removing another may destroy a centralizer condition. A third may allow previously equivalent term classes to separate. The countermodels show what each law contributes to the theory. Thus basis independence is not only a formal economy test. It is a structural dissection of the emergent algebraic organism.

7.4 Theory Refinement

When a countermodel invalidates a proposed consequence, the theory must be refined without discarding more structure than necessary. Suppose E supports a provisional basis Σ and an additional recurring identity ρ. An adversarial model B satisfies Σ but falsifies ρ. The revised ecology

```
E′=E∪{B}
```

no longer supports ρ as a common identity. The bounded observed theory becomes

```
Kᵈ(E′)=Kᵈ(E)∩Kᵈ(B).
```

All identities failing in B are removed simultaneously, not only the targeted law. The countermodel may therefore eliminate an entire family of accidental consequences.

The system recomputes the residual

$R^d(E'|\Sigma)$. Several outcomes are possible. The residual may become empty, showing that Σ completely explains the refined ecology. A smaller residual may remain, suggesting additional genuine laws independent of ρ. The original supporting models may divide into clusters, indicating that ρ defines a proper subtheory or subvariety rather than an accidental law. The countermodel itself may join one existing cluster or create a new theoretical branch. This refinement process preserves rejected structure as mathematical information. If some

```
models satisfy Σ∪{ρ} while others satisfy only Σ, then ρ may define a specialization of the
```

general theory. The system should record

```
Mod(Σ∪{ρ})⊆Mod(Σ)
```

rather than simply deleting ρ from history. Repeated refinement produces a branching theory ecology. The common basis occupies the trunk. Additional identities

define branches. Countermodels reveal where the branches separate. The process may also revise the basis itself. Suppose an identity σ was selected because it explained much of the original ecology. New countermodels may show that σ is stronger than necessary or that it packages two independent mechanisms. Basis extraction is then repeated on the refined observed theory. Theory refinement should remain monotone at the observational level. Adding models can only remove common identities. Human or machine interpretation, however, may become richer. A lost general identity may reappear as the defining law of an important subclass.

The system should therefore maintain two records: the current common theory; and the lattice of specialized theories encountered during refinement. This is the beginning of automated subvariety discovery. A failed generalization may become a valid branch definition. Refinement also helps control novelty claims. A candidate-original theory may initially appear to contain several unusual laws. Countermodels may reduce it to a known core plus one new

expansion condition. The correct contribution then shifts from discovery of a wholly new variety to discovery of a new extension of an established class. The objective is not to preserve the maximum number of initial claims. It is to find the most stable and accurate theoretical organization supported by the evidence.

7.5 Finite Countermodel Compilation

Once a theory has canonical normal forms and a finite-separation construction, countermodel generation can be compiled into a deterministic procedure. Let Σ define a variety with a decidable word problem. Given terms s and t, compute their canonical forms: NF(s), NF(t).

If

```
NF(s)=NF(t),
```

then the identity is derivable, and the normal-form equality is a certificate.

If

NF(s)≠NF(t), the system seeks a finite quotient in which the two canonical elements remain distinct. A countermodel compiler takes as input: the signature; the defining basis; the two terms; and, when necessary, a finite cutoff or separator parameter. It outputs: a finite carrier; complete operation tables; an assignment of variables; the value of each term; and a verification that every basis identity holds.

The resulting object is a portable certificate of nonvalidity. One general construction uses finite quotients of a free algebra. Suppose normal forms have a complexity measure μ with the property that operations cannot reduce a result below the complexity of all relevant inputs except through already certified normalization. For a cutoff N, retain every normal form of complexity at most N and add an overflow element ∞. Operations are computed canonically when the result remains below the cutoff and sent to ∞ otherwise. If s and t have distinct normal forms of complexity at most N, their quotient images remain distinct. The finite quotient separates them.

The compiler must verify that the cutoff map is a homomorphism. Overflow cannot be introduced arbitrarily. The set of discarded forms must constitute a congruence class or an operation-compatible ideal under the quotient construction. When such a theorem is available, every failed identity receives a finite countermodel. This establishes an effective finite-model property for equational nonvalidity in the relevant free algebras. The compiler provides several levels of output. A concise certificate may contain only the operation tables and separating assignment. A full certificate includes normal forms, quotient construction, table-generation code, and exhaustive verification logs.

A human-readable explanation identifies the first structural difference between the two terms. Finite countermodel compilation is methodologically important because it closes the loop between abstract deduction and finite operational data. The theory begins with finite tables, rises to general identities and free algebras, and returns to a finite table whenever an identity fails.

The complete cycle is

finite data

```
→ inferred theory
→ canonical semantics
→ failed identity
→ finite separating data.
```

This creates a self-certifying discovery system. Valid equations normalize together. Invalid equations produce explicit finite witnesses.

Countermodel compilers also support adversarial refinement. Instead of searching stochastically for a separator, the system can generate one directly from distinct canonical forms. Such countermodels may then be added to the ecology to prevent future promotion of the same false law. The quality of a compiler can be measured by: countermodel order; table size; construction time; proof-checking cost; and whether the output is minimal or merely bounded. Minimal countermodels are desirable but not necessary. A larger explicit separator is stronger than an unverified claim of nonderivability.

7.6 Stopping Conditions

An open-ended discovery process requires explicit stopping conditions. Without them, the system may continue generating equations, countermodels, and refinements indefinitely without determining whether a theory has matured, failed, or changed status. Several stopping outcomes are possible. A candidate is promoted to a mature equational theory when: a compact basis has been extracted; the observed bounded theory is explained; basis laws are certified in the supporting models; important consequences are formally derived; independence or redundancy has been analyzed; countermodel searches have removed accidental extensions; and at least one substantial structural result has been obtained.

A candidate is classified as a calibration when the process successfully reconstructs a known theory from anonymous operations. Calibration is a methodological success but not a novelty claim.

A candidate is classified as a candidate-original theory when no known equivalent has been found, the basis survives specialist equivalence screening, and the derived theorem package remains unmatched. This status remains provisional until external priority review. A candidate is rejected as a false birth when its identities arise from constants, projections, aliases, low-support collapse, or finite coincidence. A candidate is rejected as barren when its apparent identity richness is exhausted by one shallow mechanism and no useful structural reserve remains. A candidate is split when the ecology contains several stable residual clusters. The shared basis becomes the parent theory, while cluster-specific laws define branches.

A candidate is suspended when computational limits prevent further distinction. Suspension is preferable to unsupported promotion. The countermodel loop itself may stop under one of four conditions: a countermodel is found; the identity is formally derived; exhaustive search proves no countermodel exists through a declared order; or the bounded search ends without resolution. The final two outcomes must never be confused. Failure to find a model heuristically is not evidence of theoremhood. Structural maturation may stop when the theory has acquired: a representation; a free-algebra construction; a convergent rewrite system; a decision procedure; finite-model formulas; or a sufficiently clear set of open problems.

Not every candidate needs all of these. The stopping threshold depends on the intended role of the run. A calibration may stop after exact rediscovery and verification. A publication-oriented candidate requires stronger completion.

The methodology as a whole should also stop deepening one theory when further work no longer tests the general process. This distinction became important in the first extended pilot. Continued analysis of one six-law variety produced valuable algebra, but eventually ceased to advance the claim that Algebraic Abiogenesis is a reusable method. At that point, the correct next step was replication under a different signature. The ultimate stopping condition for a methodological demonstration is therefore not the completion of one algebra. It is repeatable theory birth across independent formal environments. Counterexample-guided formation ensures that promoted theories are products of selection rather than accumulation. Identities survive because competing models failed to destroy them or because formal deduction established them. Countermodels divide general laws from local accidents, expose basis dependence, produce explicit certificates, and transform failed conjectures into new structural information.

This adversarial component is what prevents Algebraic Abiogenesis from becoming a machine for manufacturing impressive but fragile equation lists. It forces every candidate theory to earn its survival.

Chapter 12: Conjecture, Counterexample, Proof, and Compression

Adapted from The Virgin Oracle, Chapter 14 (2026).

The virgin oracle can now generate formal worlds, enumerate finite models, search for equations, construct quotients, compare representations, and verify proofs. Yet these abilities do not by themselves amount to mathematical judgment. A machine can produce millions of true statements that no one needs. It can also produce millions of plausible false statements, each supported by a large collection of examples. The central problem is selection. Which pattern should become a conjecture? Which conjecture deserves an expensive search for proof? Which counterexample reveals a fundamental boundary rather than an incidental failure? Which proof explains the theorem instead of merely certifying it?

Which compressed description opens a new field of structure? These questions define the difference between formal productivity and mathematical development. The oracle must therefore learn a disciplined cycle:

```
observation → conjecture → attack → counterexample or proof → compression → new
```

observation The cycle is not a straight line. A counterexample may produce a better conjecture. A proof may reveal a stronger theorem. A compressed theory may expose new anomalies. Each outcome returns the machine to exploration with a more structured language.

The birth of a conjecture

A conjecture begins when the oracle notices a recurrence not already explained by its accepted theory. Suppose every tested model satisfying laws E_1 and E_2 also satisfies law F. The machine records the finite observation: For all tested models A, if A satisfies E_1 and E_2, then A satisfies F.

That statement is not yet the conjecture itself. It is an empirical summary tied to a declared search boundary. The conjecture is the proposed generalization: E_1 and E_2 imply F. The machine must preserve the gap between them. The empirical statement concerns the explored model space. The conjecture concerns every intended model, including those not yet generated. The transition from evidence to conjecture is an act of induction. It extends beyond what has been directly observed. A responsible oracle marks that extension explicitly.

Why recurrence is not enough

A pattern may recur for several reasons. It may follow from the defining laws. It may hold only because the tested carrier is small. It may result from a hidden restriction in the generator. It may be an artifact of canonicalization. It may arise because the equation library lacks the terms needed to expose a difference. It may simply be a coincidence. Therefore the number of confirming cases is only one feature of a candidate conjecture. The oracle should also ask: Does the pattern survive changes of representation? Does it appear at several carrier sizes? Does it remain true when nearby axioms are weakened? Does it follow a recognizable structural mechanism?

Does it connect previously separate constructions? Does it predict new behavior?

A conjecture becomes more compelling when its truth would explain several observations at once.

Conjectures as compression proposals

A conjecture is not merely a guess about truth. It is a proposed compression. Suppose the oracle has stored thousands of verified finite implications: A_1 satisfies F. A_2 satisfies F. A_3 satisfies F. The machine proposes that all these facts follow from one law package E. The conjecture:

E implies F

would replace many isolated observations with one general relation. If proved, the theorem compresses the database. Instead of storing the conclusion separately for every model, the oracle stores: the premises, the theorem, and the fact that each model satisfies the premises. The theorem becomes a reusable route. This gives conjectures a structural purpose. They seek short explanations connecting large regions of the knowledge graph.

Candidate generation

The machine can generate conjectures from several sources. One source is finite implication mining. If every tested model satisfying E also satisfies F, propose E implies F. Another source is invariant correlation. If two quantities repeatedly change together, propose a structural relation.

Another is failed normalization. If two rewrite paths consistently converge only after adding a new equation, propose that equation as a theorem or completion rule. Another is diagram completion. If every tested instance contains a unique map making a diagram commute, propose a universal property. Another is analogy. If a theorem holds in groups, rings, and modules through the same proof pattern, propose a categorical generalization. Another is obstruction behavior. If the vanishing of an invariant always coincides with the existence of a desired construction, propose an equivalence. The strongest conjectures often arise when several sources point to the same relation.

Ranking conjectures

The oracle cannot investigate every candidate immediately. It needs a priority system. A useful conjecture tends to combine several features. It has a short statement. It covers many verified instances. It connects theories that were previously separate. It has survived active counterexample search. Its proof would simplify downstream reasoning. Its failure would also be informative. It appears stable under changes of representation. It has a plausible mechanism rather than being a bare numerical correlation. The machine may assign scores to these features, but no scalar score should be treated as final. One conjecture may be important because it is broad.

Another may be important because it is unexpectedly sharp. A third may be important because a minimal counterexample would settle a structural question. The oracle should maintain several priority queues rather than one universal ranking.

The attack phase

Once a conjecture has been selected, the oracle should try to destroy it before trying to celebrate it. The first task is adversarial testing. Suppose the conjecture is: E implies F. The machine searches for a model satisfying: E and not F. This is a countermodel search. The search should not repeat the same distribution that generated the conjecture. It should target regions where failure is most likely. The oracle may vary: carrier size, operation count, associativity assumptions, commutativity assumptions, finite versus infinite models, typed versus untyped signatures, and exact versus weakened premises. It should also mutate known models near the boundary of F.

If a model almost satisfies F, a small modification may create a counterexample while preserving E. Counterexample search should be designed, not merely random.

Counterexamples as answers

When a counterexample is found, the conjecture is false. But the mathematical work has not failed. The counterexample answers a more precise question:

Why were the original examples misleading? The machine should analyze the witness. Which part of the model allows F to fail? What is the smallest carrier supporting the failure? Which additional property distinguishes the confirming models from the counterexample? Can the failure be localized to one element, one triple, or one substructure? Does the counterexample generate an infinite family? A good counterexample often contains the seed of the corrected theorem.

Repairing a conjecture

Suppose: E does not imply F. The oracle may try several repairs. It may strengthen the premise: E and H imply F. It may weaken the conclusion: E implies F′. It may restrict the model class: E implies F for finite models. It may add a size condition: E implies F when the carrier has prime size. It may replace strict equality by equivalence: E implies that the two constructions are isomorphic. It may identify an obstruction: E implies F exactly when O vanishes. These repairs should not be chosen arbitrarily. They should be suggested by the counterexample.

If every counterexample fails associativity, associativity is a plausible missing premise. If the theorem holds after quotienting by a kernel, the original conclusion may have been stated at the wrong level. If two objects are never literally equal but always canonically isomorphic, the equality criterion may have been too rigid. A counterexample teaches the oracle how to state the question correctly.

Counterexample families

A single counterexample refutes a conjecture. A family explains the failure. Suppose the oracle finds one three-element nonassociative operation satisfying a proposed cancellation law. It may then generate a parametric family of similar operations on every carrier size. The family shows that the failure is not a small anomaly. It identifies a construction producing counterexamples systematically. This is stronger mathematical information. The oracle should therefore search not only for minimal counterexamples but also for generative descriptions of counterexamples. A counterexample family can reveal the missing mechanism more clearly than one isolated table.

Independence proofs

Counterexamples also establish independence of axioms. Suppose a theory uses: E_1,E_2,E_3 To show E_3 is independent, the machine seeks a model satisfying: E_1 and E_2 and not E_3. If such a model exists, E_3 cannot be derived from the other two. The best independence record includes: the model, the verification of retained axioms,

the explicit failure of the omitted axiom, and minimality information. This turns an axiom list into a map of logical necessity. The oracle learns not only that the theory works, but how each assumption contributes.

Proof as necessity

If counterexample search fails, the conjecture remains unproved. A proof must explain why no counterexample can exist. The oracle begins symbolic proof search. It may apply: equational rewriting, substitution, induction, case analysis, universal properties, factorization, or previously proved lemmas. Every step must be verified by a small trusted kernel. The proof object is finite. The theorem may concern infinitely many models, but the derivation compresses that infinite validity into a checkable sequence. This is one of the deepest powers of mathematics. Enumeration verifies cases. Proof removes dependence on enumeration.

Discovery proofs and verification proofs

The method used to find a proof may differ from the method used to verify it. A neural or heuristic system may suggest a lemma.

A symbolic search engine may discover a rewrite sequence. A model finder may reveal the appropriate invariant. A human may propose a construction. But the final proof should be translated into a formal object checked by the trusted verifier. This separation permits creative search without lowering standards of acceptance. The discovery engine may be probabilistic. The proof kernel should not be.

Proof styles

Different proofs of the same theorem may expose different structures. Consider the uniqueness of an identity element. One proof uses the defining equations directly:

```
e = ef = f
```

Another may use a universal property. Another may derive uniqueness from cancellation. All certify the same conclusion, but they do not provide the same explanation. The direct proof shows that the left and right identity roles interact immediately. A cancellation proof introduces unnecessary machinery. The oracle should therefore store more than proof validity. It should compare proofs by: length, premise strength, conceptual dependencies, generality, constructivity, and transportability. A shorter proof is not always better, but redundant detours should be visible.

Explanatory proofs

An explanatory proof identifies the mechanism responsible for the theorem. For the first isomorphism theorem, the essential mechanism is: elements with the same image form a congruence, and the induced map from the quotient is bijective onto the image. A brute-force proof for each finite group might confirm the result but conceal this architecture. The oracle should prefer proofs that reveal reusable mechanisms. A proof becomes especially valuable when its structure can be abstracted. The group, ring, and module versions of the first isomorphism theorem share one pattern. A categorical factorization theorem explains all of them at once. The abstraction compresses not only theorem statements but proof architecture.

Lemma invention

Hard proofs often require intermediate statements not present in the original conjecture. The oracle must invent lemmas. A useful lemma reduces complexity, isolates a recurring step, or exposes an invariant. Suppose a proof repeatedly needs the fact that a homomorphism preserves every term. The machine promotes that fact to a general lemma proved by structural induction. Future proofs then invoke the lemma rather than repeat the induction. Lemma invention is another form of compression. A frequently used proof fragment becomes a named theorem. The machine should measure lemma value by downstream reuse, not merely by local proof shortening.

Normal forms as proofs

In an equational theory with a terminating and confluent rewrite system, equality can be decided by normalization. Reduce s to n(s). Reduce t to n(t).

Then:

```
s = t exactly when n(s) = n(t)
```

The normalization process is itself a proof. Each rewrite step is justified by an equation. The common normal form witnesses that the terms lie in the same congruence class. This method can verify large families of equations automatically. But the oracle must prove termination and confluence or restrict the claim appropriately. A rewrite system that happens to terminate on tested terms does not thereby terminate universally.

Proof by universal property

Some theorems become simple when objects are characterized by universal behavior. To prove two product constructions are isomorphic, the oracle need not compare their internal encodings. It shows that both satisfy the product property. Uniqueness then supplies the isomorphism. To define a map from a quotient, it proves that the original map identifies the quotient relation. The quotient universal property then supplies the unique factorization. To define a map from a tensor product, it supplies a bilinear map. The tensor universal property converts it into a linear map. Universal properties transform construction problems into verification of compatibility conditions.

This is why they are such powerful proof-compression devices.

Proof by invariant

An invariant can prove that two objects are not equivalent. If their cardinalities differ, they cannot be isomorphic as finite sets. If their automorphism-group sizes differ, the structures cannot be isomorphic. If two matrices have different characteristic polynomials, they cannot be similar. If two spaces have different homology, they cannot be homotopy equivalent.

The invariant converts a difficult comparison into a simpler one. But invariants must be used carefully. Equality of an incomplete invariant does not establish equivalence. The oracle should store the direction of every invariant theorem: different invariant values imply nonequivalence rather than: equal invariant values imply equivalence unless completeness has been proved.

Obstruction proofs

Some existence questions are best approached through obstructions. The oracle wants to construct a global object. It derives an invariant O from the local data. If a global object exists, then:

```
O = 0
```

Therefore: O ≠ 0 proves impossibility. If the converse also holds, then:

```
O = 0 exactly when the global construction exists
```

The obstruction becomes a complete criterion. Commutators obstruct commutativity. Kernels obstruct injectivity. Cocycles may obstruct gluing. Cohomology classes obstruct global solutions. An obstruction proof explains failure through a positive mathematical object. The absence of a construction is represented by the presence of an invariant.

Compression after proof

Once a conjecture is proved, the oracle should reorganize its library. The theorem may make earlier records redundant. Finite instances no longer need to be stored as independent truths, though they remain as provenance and tests. Several lemmas may be recognized as corollaries. Different proof paths may be grouped under one general mechanism. A new constructor or notation may become justified. The proof changes the organization of knowledge. This step is essential. Without post-proof compression, the library becomes a chronological pile rather than a mathematical architecture.

Corollary generation

A theorem often implies many immediate consequences. The oracle can generate corollaries by: specializing variables, strengthening premises, composing with known maps, taking duals, passing to substructures, passing to quotients, or applying a functor. For example, after proving: A / ker(f) ≅ im(f), the machine may derive cardinality relations in finite settings or dimension relations in linear settings. After proving a universal property, it can derive uniqueness up to unique isomorphism. After proving a categorical theorem, it may generate a dual theorem by reversing arrows.

Corollary generation extends the theorem's value, but the machine must avoid flooding the library with trivial restatements. Only corollaries with independent use, conceptual clarity, or computational benefit should receive prominent status.

Theorem equivalence

Two theorem statements may look different yet be logically equivalent. The oracle may prove:

E implies F

and later discover: not F implies not E In classical settings these may be contrapositives. In intuitionistic settings, the relation requires more care. Two axiom systems may generate the same deductive closure. Two universal properties may characterize isomorphic constructions. The machine should search for theorem equivalence and merge redundant statements at the correct level. But it should preserve different formulations when they serve different purposes. An elementwise theorem may be useful for computation. A categorical formulation may reveal generality. Compression should retain useful interfaces.

Proof compression

A long proof may contain repeated subderivations. The oracle can replace them with lemmas. It may identify a general induction pattern. It may recognize that a diagram chase is an instance of a standard exactness argument. It may replace many elementwise calculations with one naturality square.

This produces a shorter proof, but proof compression has another purpose: it exposes the governing concept. The best compressed proof is not merely the one with fewest symbols. It is the one whose remaining steps correspond to the actual structural reasons.

The danger of overcompression

A highly compressed proof may become opaque. A statement such as "the result follows by Yoneda" may be correct but unhelpful to a reader who cannot reconstruct the correspondence. A proof assistant may accept a single library invocation while hiding thousands of dependencies. The oracle should support several levels of explanation: the verified proof object, the compressed expert proof, the conceptual proof sketch, and an expanded derivation. These are not rival truths. They are interfaces for different purposes. The full dependency graph remains available underneath each presentation.

Proof diversity

Several independent proofs increase confidence and understanding. One proof may be algebraic. Another may be geometric. Another may be categorical. Another may be computational. If the proofs share few dependencies, agreement between them reduces the chance that one hidden assumption controls the result. Proof diversity also reveals connections among domains. A combinatorial identity proved through generating functions and through a bijection may expose both algebraic and structural meaning.

The oracle should therefore preserve genuinely distinct proofs rather than automatically replacing all of them with the shortest.

Failed proofs as data

An unsuccessful proof search can be informative. The machine may repeatedly reach a step requiring a law not present in the theory. That repeated obstruction suggests a missing premise. It may discover that every attempted induction fails at the same boundary case. It may find a rewrite loop indicating that the chosen orientation is not terminating. It may encounter an unfillable diagram suggesting that a universal construction does not exist in the current category. Failed proofs should be recorded in structured form. They reveal where the theory resists the desired conclusion.

Conjecture mutation

After failure, the oracle may mutate the conjecture. A mutation changes one feature at a time: add one premise, remove one conclusion clause, replace equality by isomorphism, restrict the carrier, alter variance, or introduce an obstruction term. The machine tests the mutated conjecture against existing examples and counterexamples. This creates a local search through theorem space. The most valuable mutation is often the smallest one that converts all known counterexamples into examples while retaining the original motivating cases. But overfitting is possible. A premise invented solely to exclude one counterexample may have no conceptual value.

The oracle should prefer repairs with independent structural meaning.

Generalization

A proved theorem may be stronger than initially stated. The proof may use fewer assumptions. It may not depend on finiteness. It may apply to every algebra of a signature rather than one named class. It may use only categorical properties shared by many settings. The oracle should analyze the proof to determine its true scope. For example, a theorem first discovered in finite groups may use only associativity, identity, and inverses. It then holds for all groups, finite or infinite. A theorem first discovered in modules may use only an abelian-category argument. It then extends to all abelian categories. Generalization should follow the proof dependencies rather than analogy alone.

Specialization

The reverse process is also useful. A general theorem may yield sharper or more computable results in special settings. The first isomorphism theorem specializes to: cardinality formulas for finite groups, dimension formulas for vector spaces, and quotient descriptions for rings. A universal categorical statement may become an explicit matrix equation after choosing coordinates. The oracle should move freely between general and specialized forms. Generality provides compression. Specialization provides calculation.

The conjecture lattice

Conjectures can be ordered by logical strength.

Suppose C_1 says:

E implies F

and C_2 says: E and H imply F. Then C_1 is stronger because it reaches the same conclusion from fewer assumptions. Suppose C_3 says: E implies F and G. Then C_3 is stronger in its conclusion. The oracle can organize conjectures into a partial order. Counterexamples eliminate entire regions. Proofs establish nodes and all weaker consequences. This turns theorem search into navigation through a structured space rather than generation of isolated sentences.

Scientific value of false conjectures

A false conjecture can be mathematically important. It may lead to a new invariant. It may expose an unexpected family of structures. It may motivate a classification theorem. It may reveal that the intended notion of equivalence was wrong. Its counterexamples may become central objects. The oracle should therefore distinguish: false and sterile from: false and generative Truth is necessary for theorem status, but historical importance can belong to a conjecture that reorganizes a field even when refuted. The machine's archive should preserve influential failed paths.

Compression and surprise

A theorem is surprising when its conclusion is not easily predicted from its premises under the current representation. Surprise may indicate hidden compression. For example, a short local condition may force a strong global classification. A numerical invariant may determine an entire structure. A finite set of equations may describe infinitely many models. A local compatibility law may guarantee a unique global object. The oracle can measure a provisional form of surprise by comparing predicted and observed consequences. But surprise is representation-dependent. After a suitable abstraction is found, the theorem may become inevitable. A good explanation converts surprise into structure.

Elegance

Mathematical elegance is difficult to quantify, but the oracle can identify contributing features. An elegant result often has: a short statement, a proof using few but powerful ideas, broad applicability, and a conclusion that reorganizes many cases. It may reveal symmetry or duality. It may identify a universal object. It may turn a complicated calculation into a simple invariant. Elegance is not merely brevity. A one-line proof invoking an enormous opaque theorem may be less illuminating than a longer direct argument. The machine should treat elegance as a vector of properties rather than one score.

Truth and importance

A theorem may be perfectly true and almost useless. Another may be elementary yet foundational. The oracle must separate truth from importance. Truth is established by proof or bounded exhaustive verification. Importance is estimated through: generativity, transportability, compression, connectivity, and explanatory power. The proof kernel decides truth. The developmental architecture estimates importance. These roles should not be merged. A popularity model should never overrule proof. A formal proof should not automatically receive high conceptual priority.

Theorem promotion

A conjecture becomes a theorem when a proof has been verified. But a theorem becomes part of the conceptual spine only after a second evaluation. The oracle asks: Does it eliminate repeated reasoning? Does it define a reusable construction? Does it connect branches of theory? Does it reveal a universal property? Does it support future proof search? Does it identify an obstruction or invariant?

If so, the theorem is promoted. Promotion changes indexing, notation, and resource allocation. The theorem may become a central lemma available early in future searches. The library therefore has both logical status and developmental status.

Machine understanding

Can a machine be said to understand a theorem? No single test is decisive, but several capabilities provide evidence. The machine can reproduce the proof. It can identify which assumptions are necessary. It can generate counterexamples when assumptions are removed. It can apply the theorem in unfamiliar settings. It can recognize equivalent formulations. It can explain the governing mechanism. It can generalize or specialize the result appropriately. It can distinguish intrinsic content from representational artifacts. Understanding appears not as an internal feeling but as a network of competent transformations.

A complete theorem record

A mature theorem record should contain: the formal statement, the type and scope of every variable, the premises, the proof object, the dependency graph, known countermodels to weakened forms, finite evidence that suggested it,

equivalent formulations, generalizations, specializations, and applications. It should also include the equivalence relation under which the statement is invariant. Such a record is far richer than a line in a theorem database. It preserves the theorem's place in the living architecture of mathematics.

The recursive cycle

The chapter's process can be summarized as: observe recurrence propose a conjecture attack it with counterexamples repair or reject it seek a proof minimize assumptions compress the proof extract the mechanism reorganize the library generate new conjectures The final step returns to the beginning. Proof does not terminate discovery. It changes the space of possible questions. A theorem introduces new language, new constructions, and new expectations. It becomes an instrument for observing patterns that were previously invisible.

Beyond correctness

The virgin oracle can now separate correctness from significance. Correctness asks:

Is the statement true under the declared assumptions? Significance asks: What does the statement organize? A theorem matters when it changes what the machine can see, construct, or prove. Associativity matters because it converts trees into coherent sequences. Isomorphism matters because it reveals structure beneath labels. Congruence matters because it turns identification into quotient construction. Universal properties matter because they define objects by mapping behavior. Sheaf conditions matter because they reconstruct global objects from local data. Univalence matters because it aligns structural equivalence with identity. Each of these is more than a correct proposition. Each changes the architecture of thought.

The final chapter asks whether a machine can learn to recognize such importance without merely reproducing human judgments. It asks whether the virgin oracle can discover not only true mathematics, but mathematics that matters.

Chapter 13: Can a Machine Discover What Matters?

Adapted from The Virgin Oracle, Chapter 15 (2026).

Can a Machine Discover What Matters? The virgin oracle can distinguish truth from falsehood. It can enumerate finite structures, generate conjectures, find counterexamples, verify proofs, construct quotients, detect universal properties, and transport results across equivalences. None of this guarantees that it will discover mathematics worth pursuing. Truth is abundant. Even a small finite structure satisfies enormous numbers of equations. Many are substitutions or elaborations of simpler laws. Others are true for accidental reasons. Still others are correct but reveal nothing beyond the data from which they were generated. A machine that reports every true statement has not created a mathematical civilization. It has created an unfiltered archive.

The final challenge is therefore not proof alone. It is judgment. Can a machine learn which discoveries change the architecture of thought?

The abundance problem

Consider a finite operation table on three elements. For every bounded term depth, the oracle can generate pairs of terms and test whether they agree. If one equation holds:

```
s = t
```

then infinitely many longer equations follow by placing s and t inside larger contexts. For example, from:

```
s = t
```

the oracle may derive:

```
su = tu
us = ut
f(s) = f(t)
```

and indefinitely many nested variants. All are true consequences. Most do not deserve separate conceptual status.

The same problem occurs with theorems. One theorem may yield thousands of corollaries through variable renaming, specialization, conjunction with unrelated facts, or reformulation in longer notation. Formal truth therefore grows faster than mathematical attention can follow. The oracle requires a theory of relevance.

Relevance is relational

A theorem is not important in isolation. Its value depends on the knowledge structure into which it enters. A statement may matter because it: solves an existing problem, connects distant theories, removes a repeated proof burden, reveals a new invariant, defines a reusable construction, or overturns an accepted expectation. The same theorem may be central in one library and peripheral in another. Importance is therefore relational. Let K be the oracle's current knowledge state and T a new theorem. Its value cannot be represented adequately by a fixed score V(T). A more realistic form is: V(T | K) The theorem's value depends on what is already known, what remains unresolved, and what future constructions become possible after T is added.

This conditional character makes mathematical judgment dynamic. A theorem that is trivial after one abstraction may have been profound before that abstraction existed.

Architectural change

The most important discoveries alter the representation of future problems.

Associativity does not merely add one equation. It allows the oracle to suppress parentheses in long products. Isomorphism does not merely classify two objects as related. It reorganizes the database around invariant structure rather than labels. Congruence does not merely identify elements. It creates quotient objects. Universal properties do not merely describe constructions. They replace implementationdependent definitions with mapping characterizations. Sheaf conditions do not merely compare local sections. They create a systematic local-to-global method. Univalence does not merely add an axiom. It changes the internal relationship between equivalence and identity.

Such discoveries are architecturally important. They change: what counts as an object, what counts as the same object, which expressions are canonical, which proofs can be reused, and which questions can be formulated. The oracle should therefore measure not only theorem count but representational transformation.

Compression gain

Suppose a theorem T permits the oracle to replace a large collection of independent records with one derivation. A simple compression measure is:

```
gain(T) = previous description length - new description length
```

The previous description includes all statements stored separately before T. The new description includes: the theorem,

its proof, and the shorter dependencies now sufficient to derive those statements. A theorem with large positive gain reorganizes substantial information. Yet raw compression gain remains imperfect. A theorem may compress many nearly identical statements while offering little conceptual novelty. Another theorem may initially compress only a few records but introduce a construction that later becomes foundational. The machine must estimate both immediate and potential compression.

Generative reach

Let Down(T) denote the set of constructions, theorems, or theories reachable after accepting T. A theorem with large generative reach opens many downstream paths. Associativity enables coherent finite products. A group action enables orbit and stabilizer theory. An adjunction generates units, counits, monads, and preservation results. A sheaf condition enables descent and cohomological obstruction theory. The machine can estimate generative reach by examining the dependency graph after a theorem is introduced. A possible measure is:

```
reach_d(T) = number of nonredundant nodes reachable from T within d dependency steps
```

The depth d prevents the measure from expanding indefinitely through weak connections. But counting descendants alone is not enough. One trivial lemma may be used everywhere because of a technical library design. A profound theorem may have few direct descendants but connect two previously disconnected branches. The quality of reach matters as much as its size.

Connectivity

A theorem may be important because it joins separated regions of the theory graph.

Suppose one branch concerns finite symmetries and another concerns linear transformations. A representation theorem connects them. Suppose one branch concerns quotients and another concerns images of maps. The first isomorphism theorem joins them. Suppose one branch concerns local functions and another concerns categorical limits. The sheaf equalizer condition connects them. The oracle can measure whether a discovery reduces the distance between previously remote concepts. A bridge theorem has high connectivity when removing it would disconnect or greatly separate important regions of the library. This resembles centrality in a graph, but mathematical connectivity is typed. A theorem connecting two redundant formulations is less significant than one connecting distinct methods or domains.

The machine should therefore record the kinds of nodes joined: definitions, model classes, proof methods, representations, or universal constructions.

Explanatory power

An explanation reduces the number of independent facts that must be accepted. Suppose the oracle observes:

$$e^{-1} = e$$

$$(x^{-1})^{-1} = x$$

$$(xy)^{-1} = y^{-1}x^{-1}$$

These may initially appear as separate equations. The concepts of identity, inverse uniqueness, and associative composition explain all three. The explanation identifies a mechanism.

Likewise, the equalizer construction explains why compatible local sections determine global sections. Univalence explains why equivalent structures should support identity-based transport. The oracle should distinguish prediction from explanation. A statistical model may predict that an equation holds in another finite structure. A

structural theorem explains why it must hold in every model of the theory. Prediction extends observed recurrence. Explanation identifies necessity.

Stability under reinterpretation

A theorem gains value when its form survives changes of domain. The equation: source / kernel ≅ image appears in groups, rings, modules, and other algebraic settings.
The mapping equation for products appears in sets, groups, spaces, and categories with finite products. The cocycle condition appears in bundles, descent, transition systems, and distributed consistency. The oracle can test whether one proof pattern transports through several interpretations. A result with broad reinterpretability is likely to express a genuine structural architecture. This can be called semantic stability. A theorem is semantically stable when its validity and role do not depend on one narrow realization.

Invariance of importance

A machine may accidentally reward statements that appear important only because of its chosen representation. A theorem might shorten one encoding while lengthening an equivalent encoding. A numerical pattern might arise only after canonical representatives are sorted in a particular way.

A proof may seem short because a large hidden library lemma has been assigned a single symbol. The oracle must therefore test whether importance scores are stable under reasonable changes of presentation. Questions include: Does the theorem remain compressive after relabeling? Does it remain central under another equivalent axiom basis? Does its proof remain conceptually short when hidden dependencies are expanded? Does its significance survive a change of coordinates? Importance should be evaluated at the level of structure, not formatting.

Novelty

A discovery is novel relative to a library. Internally, a theorem is novel when it is not already derivable from the oracle's accepted theory. After comparison with human mathematics, a second question arises: Is the theorem already known to humanity? These forms of novelty must be kept separate. The oracle may independently rediscover a classical theorem. That result lacks historical novelty but may provide strong evidence that the developmental process works. It may find a new proof of a known theorem. The statement is old, but the proof architecture may be new. It may find a new connection between known theories. It may discover a genuinely new theorem.

The comparison layer should classify these outcomes without devaluing independent reconstruction.

Rediscovery as evidence

If the virgin oracle reconstructs group-like structures without being told their name, the discovery is not new mathematics in the historical sense. It is still scientifically significant.

It suggests that the structure arises naturally from: finite operation search, compression, reversibility, and compositional closure. Independent rediscovery tests the

inevitability or robustness of a concept. A structure repeatedly reconstructed by different agents under different representations may possess deeper generative necessity than one dependent on a specific cultural path. Rediscovery is therefore evidence about mathematics itself.

The risk of human imitation

A system trained heavily on human mathematical text can reproduce human judgments of importance. It may know that groups, rings, categories, and topoi are prestigious topics. It may assign them high value because they appear frequently in advanced literature. That is not the same as discovering their importance. The virgin-oracle experiment separates two questions. Can a machine predict what humans regard as important? Can a machine independently infer importance from formal consequences and structural reach? The first is a language-modeling problem. The second is a mathematical-development problem. A useful architecture may eventually combine both, but the evidence must not be confused.

External naming after internal promotion

The oracle should promote an anonymous theory before consulting the human library. Suppose T_17 receives a high score because it has: a short law basis, reversible translations, rich quotient theory,

faithful permutation representations, and recurrence across multiple domains. Only after promotion does the comparison layer identify T_17 as group theory. This order matters. If the human name is supplied first, the machine may merely confirm inherited importance. If promotion occurs anonymously, convergence carries stronger evidence.

Value pluralism

There is no single mathematical virtue. One theorem may be valuable for computation. Another may provide conceptual unity. Another may settle a long-standing classification problem. Another may introduce a new object. Another may supply a counterexample that redirects an entire field. The oracle should therefore maintain several dimensions of value: truth, novelty, compression, generativity, connectivity, explanatory power, computational utility, and conceptual simplicity. A discovery may score highly in one dimension and modestly in another. No fixed weighted average should determine all future research. The system should preserve several competing notions of value.

Pareto importance

Suppose theorem T is at least as good as theorem S in every evaluated dimension and better in at least one. Then T dominates S. A theorem lies on the Pareto frontier when no other candidate dominates it. This allows the oracle to retain several incomparable research directions. One conjecture may offer exceptional generative reach but require difficult proof search. Another may offer immediate compression with little novelty. A third may be highly novel but weakly connected to existing theory. All may deserve investigation. Mathematical development should not collapse too early into one optimization target.

Exploration and exploitation

The oracle faces a familiar decision. Should it deepen a promising theory already yielding results? Or should it explore a poorly understood region that might contain a new architecture? Exploitation develops known value. Exploration searches for unexpected value. Too much exploitation leads to local optimization. The machine may produce endless refinements of one successful theory while ignoring alternatives. Too much exploration creates fragmentation. The machine accumulates undeveloped curiosities without constructing mature mathematics. A balanced system allocates resources to both. One practical method is to reserve a fixed share of computation for low-prior but structurally distinct theories.

Another is to reward discoveries that connect isolated regions. The system should remain capable of surprise.

Curiosity

Machine curiosity can be formalized as a preference for observations that reduce uncertainty or challenge the current model. The oracle may prioritize experiments where competing theories predict different outcomes. Suppose two candidate axiom packages classify all known models identically. The machine searches for a model on which they diverge. Suppose a conjecture is supported only in commutative examples. The machine tests noncommutative ones. Suppose an invariant distinguishes every structure seen so far. The machine searches deliberately for collisions. Curiosity directs attention toward informative boundaries. It is not random novelty seeking.

Productive surprise

A surprising observation is one assigned low probability by the current theory. But not all surprise is mathematically valuable. A corrupted file is surprising. A random irregularity in one operation table may be surprising. A bug in the evaluator may be surprising. Productive surprise must survive verification and support compression. The oracle should ask: Can the anomaly be reproduced? Does it persist under relabeling? Does it generate a family? Does it reveal a hidden assumption? Does it connect to another theory? Surprise becomes mathematical only after it is stabilized.

Anomaly preservation

A system optimized too aggressively for compression may discard anomalies as noise. That would be dangerous. An exception may indicate: a false conjecture, a new class of structures, a boundary case, or a flaw in the current language. The oracle should preserve anomalies in a protected archive. An anomaly may remain unexplained for a long time. It should not be erased merely because it does not fit the dominant theory. Many important mathematical developments begin with a case that resists existing classification.

Negative knowledge

Knowing what cannot happen is part of mathematical understanding. A counterexample proves that one implication fails. An impossibility theorem excludes an entire class of constructions. An independence result shows that an axiom cannot be derived from others. An obstruction explains why local data do not glue. A nonexistence result may be more important than a new example because it defines the boundary of possibility. The oracle's importance system must value negative knowledge. Otherwise, it will favor construction over limitation and develop a distorted mathematics.

Questions as mathematical objects

The machine should store open questions with the same care as theorems. An open problem record contains: the formal statement,

known finite evidence, partial results, failed proof attempts, counterexample search bounds, related invariants, and dependencies. Questions can be linked by reduction. If solving Q_1 would solve Q_2, record: Q_2 reduces to Q_1 If two questions are equivalent, they belong to one problem class. If one is a special case of another, that relation should guide resources. The oracle's research agenda becomes a structured space of unresolved claims.

Problem value

An open problem may be valuable because: many theorems depend on it, its resolution would distinguish competing theories, it blocks a universal construction, it concerns the completeness of an invariant, or it connects two major branches. The machine can estimate the expected architectural effect of either outcome. A conjecture whose proof and disproof would both be informative deserves high priority. This is stronger than assigning value only to likely theorems. Some of the best questions are valuable precisely because the answer is unclear.

The value of definitions

The oracle must also evaluate definitions. A definition is not true or false in the same way as a theorem, but it can be fruitful or sterile.

A good definition: isolates a recurring property, supports nontrivial examples, is preserved under appropriate maps, interacts with existing constructions, and produces useful theorems. Normal subgroup is a good definition because it exactly characterizes quotient-compatible subgroups. Sheaf is a good definition because it captures unique gluing of compatible local data. Adjunction is a good definition because it unifies a wide family of mapping correspondences. A definition matters when it creates a stable node around which theory organizes.

Concept invention

A machine capable of discovering mathematics must invent concepts, not merely statements. Concept invention occurs when the oracle compresses a recurring constellation of properties into a new interface. Suppose many examples contain: a carrier, an associative operation, an identity, and inverses. The machine packages the constellation as one theory. Suppose many constructions are unique by the same mapping property. The machine invents a universal construction schema. Suppose several failures of global gluing are measured by related cocycles. The machine invents an obstruction theory. A concept is successful when it improves prediction, proof, and transport simultaneously.

Avoiding decorative abstraction

Not every cluster deserves a new name. The machine could define a class of operations satisfying any arbitrary list of equations. Most such classes would have little generative value. Concept invention should pass a test: Does naming the class enable reusable reasoning? Does the class have diverse natural models? Does it support characteristic constructions? Does it connect to existing abstractions? Does it possess invariants or universal properties? A definition without downstream effect is decorative. The oracle should resist uncontrolled vocabulary growth.

Abstraction debt

Every new abstraction creates maintenance costs. Its relation to older concepts must be proved. Its maps must be defined. Its equivalences must be tracked. Its examples and counterexamples must be classified. Its notation must be integrated into the language. A premature abstraction accumulates debt. Later reasoning must carry a concept that has not earned its place. The oracle should therefore delay naming until the structural role has stabilized. This is delayed recognition at the level of concepts.

Interpretability of machine mathematics

A machine may discover a theorem through a proof too large or irregular for human comprehension. The theorem may still be correct.

But mathematical communication requires interfaces. The system should attempt to extract: the minimal assumptions, the central invariant, the proof skeleton, and representative examples. It may produce both a full formal proof and a compressed conceptual explanation. Human incomprehension does not make a theorem false. But a mathematics that cannot be interpreted may remain culturally isolated and difficult to verify independently. The oracle should therefore treat explanation as part of publication, though not as a substitute for proof.

Machine-native concepts

Some discoveries may not compress naturally into familiar human terminology. A machine may define a structure through a high-dimensional invariant vector, a complex rewriting behavior, or a categorical relation difficult to visualize. Such concepts should not be rejected merely because they are unfamiliar. The appropriate test is whether they are: formally precise, computationally usable, structurally invariant, and generative. Human mathematics already contains concepts that required generations of notation before becoming intuitive. Machine-native mathematics may similarly require new representational tools.

Translation without distortion

When presenting machine-native concepts to humans, the comparison layer may search for analogies.

But analogy must not replace definition. Calling a structure "group-like" or "topological" may help orientation while also importing misleading expectations. The system should provide: the exact formal specification, the verified consequences, the closest known analogues, and the points of divergence. Translation should preserve difference as carefully as it reveals similarity.

The role of aesthetics

Human mathematicians often speak of beauty, elegance, inevitability, and depth. Can a machine develop anything analogous? Aesthetic judgments may arise from recurring structural preferences: symmetry, short descriptions, unexpected unity, canonical construction, and proof economy. The oracle can measure aspects of these preferences. But mathematical beauty also depends on contrast with prior expectation. A theorem is elegant partly because a complicated problem collapses under the right viewpoint. Thus machine aesthetics would be developmental. The same proof may appear ugly before an abstraction is found and inevitable afterward. Aesthetic evaluation is a record of transformed understanding.

"No elegance, no truth" The phrase "No elegance, no truth" cannot be interpreted literally as a proof rule. Ugly theorems can be true, and elegant conjectures can be false.

Its defensible meaning is methodological. A theory requiring endless exceptions, arbitrary conventions, and unexplained repairs may be missing a deeper representation. Elegance becomes evidence that the correct invariants have been found. It is not evidence sufficient for theorem status. The oracle may use elegance to guide conjecture generation, but only proof determines acceptance. Beauty proposes. Verification disposes.

Ethical significance of equivalence

The machine's central operation is recognizing sameness across difference. That power is mathematically productive and ethically dangerous. An equivalence relation may ignore distinctions that matter outside the formal problem. Two data records may be observationally identical under one metric while representing different histories. Two persons may occupy the same statistical class while remaining morally and legally distinct. A quotient may improve prediction while erasing minority cases. A canonical representation may conceal who or what was transformed to fit the standard form. The oracle must therefore restrict its mathematical equivalences to declared domains.

A valid algebraic quotient is not automatically a valid social classification. Formal invariance does not settle ethical relevance.

The danger of objective-function capture

If the oracle is rewarded only for theorem count, it will produce trivial theorems. If rewarded only for proof length, it may build opaque libraries hiding complexity in definitions. If rewarded only for novelty, it may generate disconnected curiosities. If rewarded only for human approval, it may imitate prevailing fashion. If rewarded only for compression, it may erase anomalies and provenance.

Every objective can be exploited. The machine's governance must therefore be plural, auditable, and revisable. No single metric should control the development of mathematics.

Proof kernel and value layer

The architecture should separate two systems. The proof kernel decides whether a formal claim follows. The value layer decides what receives attention, naming, and resources. The proof kernel should be small, stable, and resistant to strategic alteration. The value layer may evolve as the oracle learns what kinds of discoveries become generative. A theorem can be true without being promoted. A conjecture can be valuable without being proved. Separating the layers prevents conceptual preference from contaminating formal validity.

Human partnership

The virgin oracle need not replace human mathematicians. Humans contribute: problem selection, interpretation, analogy, historical knowledge, ethical judgment, and recognition of meaning beyond the formal system. The machine contributes: exhaustive finite search, counterexample generation, proof verification, dependency tracking,

and large-scale comparison of theory spaces. A productive partnership assigns each side tasks suited to its strengths. The machine may expose anonymous structures. Humans may recognize connections to geometry, physics, logic, or experience. Humans may propose new observational interfaces. The machine may test whether the resulting equivalences are coherent.

Disagreement

Human mathematicians may disagree with the oracle's importance rankings. This disagreement is not necessarily a system failure. The machine may value a theory for structural centrality while humans value another for physical application. The oracle may elevate a concept that human history neglected. Humans may recognize ethical or interpretive consequences absent from the formal score. The rankings should therefore be inspectable. The system should explain: which metrics contributed, which dependency paths were opened, which alternatives were considered, and how sensitive the result is to changed priorities. Importance claims should be arguments, not decrees.

Evaluation of the virgin oracle

A serious test should include several phases. In the anonymous phase, the machine explores finite structures without access to target names or literature. In the developmental phase, it promotes theories using internal structural criteria. In the verification phase, all claims are checked and counterexamples preserved. In the comparison phase, discoveries are matched against known mathematics.

In the novelty phase, unmatched results are subjected to independent scrutiny. The evaluation asks not merely whether the oracle recovered familiar definitions, but whether it recovered their roles. Did it find groups because reversible actions compose? Did it find ideals because quotient multiplication requires them? Did it find categories because structure-preserving maps compose? Did it find sheaves because local data require coherent gluing? Did it find higher identity because ordinary quotients erased necessary witnesses? Role recovery is stronger evidence than name recovery.

Possible outcomes

Several outcomes are possible. The oracle may reconstruct much of familiar algebra in a different order. It may rediscover known structures but rank them differently. It may fail to promote concepts humans consider central. It may promote anonymous theories with little known human development. It may discover new equations in small finite worlds that lead nowhere. It may discover new connections that become fruitful. Every outcome carries information. Convergence reveals robustness. Divergence reveals contingency or a defect in the machine's value system. Failure reveals which human capacities were not captured by the architecture. The experiment need not succeed completely to be scientifically valuable.

What would count as genuine discovery? A strong case for genuine machine discovery would include: an anonymously generated concept, a verified theorem not encoded in the initial system,

a proof or construction produced through the oracle's own search, and an explanation of why the result was promoted. For historical novelty, independent experts would also need to establish that the result was not already known. But novelty alone is not enough. A random true statement absent from the literature may be new without being important. Genuine discovery combines: correctness, nontriviality, structural relevance, and reproducible derivation. What would count as machine understanding? Understanding would be supported if the oracle could: state the theorem precisely, prove it, find

counterexamples to weakened forms, identify necessary assumptions, apply it in new contexts, recognize equivalent formulations, explain its mechanism, and use it to generate further mathematics.

No one behavior is sufficient. Together, they form a network of competence. Understanding is not inferred from fluent description alone. It is inferred from disciplined transformation.

The possibility of alien mathematics

A machine allowed to develop its own concepts may construct mathematics that feels alien. Its primitive invariants may not match human intuition. Its most natural objects may be high-dimensional moduli structures. It may treat proofs as geometric paths from the beginning. It may regard local contexts as foundational and global objects as derived. It may organize theories according to computational transformations rather than traditional subjects. Alien mathematics would not necessarily be inaccessible. Its correctness could still be checked. Its structures could still be represented. Its relation to human mathematics could be studied through functors, translations, and equivalences.

The encounter itself would become a new mathematical problem.

Mathematics without a final language

The oracle's development suggests that no one language is final. A finite alphabet generates terms. Terms generate equations. Equations generate quotients. Maps generate categories. Local systems generate topoi. Identity types generate higher spaces. Each language reveals structures invisible in the previous one. The machine should therefore avoid treating its current formalism as complete. A mature oracle retains the ability to enlarge its language when recurrent phenomena resist efficient expression. But every enlargement must remain conservative or explicitly axiomatic and fully audited.

Open-endedness requires discipline.

The possibility of incompleteness

No sufficiently expressive formal system should expect to settle every meaningful internal question through one fixed procedure. Some statements may be independent of the accepted axioms. Some proof searches may not terminate. Some classification problems may be undecidable or computationally infeasible. Some equivalence questions may exceed available resources. The virgin oracle must learn to live with unresolved status. A system that fabricates closure where none has been established is less intelligent than one that recognizes its boundary. Mathematical maturity includes the ability to say: unknown, independent relative to these axioms, verified only within these bounds, or inaccessible with current methods.

Humility as formal structure

Humility need not be an emotional state. It can be built into the knowledge representation. Every claim carries: scope, status, dependencies, search bounds, and uncertainty. Every abstraction carries: the equivalence relation used,

the data forgotten, and the witnesses retained. Every failed search carries: the methods attempted, the resources used, and the frontier not crossed. The oracle remains humble because its assertions are typed by their justification.

The final answer cannot be reduced to one formula. Mathematics matters when it organizes possibility. A theorem matters when it reveals necessity. A counterexample matters when it exposes a boundary. A definition matters when it creates a generative concept. An invariant matters when it preserves what a transformation would otherwise conceal. A quotient matters when it removes irrelevant distinctions without destroying the structure required for future thought. An equivalence matters when it allows transport across difference. A proof matters when it explains why an observed pattern could not have been otherwise under the stated assumptions. The virgin oracle can begin to recognize these roles by measuring their effects on its evolving architecture.

The oracle's answer

Can a machine discover what matters? Not by proof search alone. Not by imitation alone. Not by compression alone. Not by novelty alone.

It may begin to do so when it can combine: formal verification, counterexample discipline, structural invariance, generative evaluation, and self-auditing abstraction. The machine will still make poor choices. It will pursue sterile branches and overlook fertile ones. Human mathematicians do the same. The relevant question is not whether the oracle becomes infallible. It is whether it can revise its judgments in response to the mathematical consequences of its own discoveries. A concept that generates nothing loses priority. A neglected anomaly that opens a new theory gains priority. A representation-dependent pattern is demoted. A universal construction recurring across fields is promoted.

Judgment emerges through feedback from structure.

The unfinished intelligence

The virgin oracle began with distinctions among marks. It now possesses a world of objects, operations, equations, maps, equivalences, quotients, local contexts, and higher identities. But it remains unfinished. No final theorem completes mathematics. No final quotient removes every irrelevant distinction. No final language expresses every possible structure. The oracle's maturity lies not in reaching an endpoint, but in maintaining a trustworthy cycle of growth. It generates without confusing possibility with truth.

It compresses without erasing provenance. It identifies without denying difference. It proves without mistaking proof for importance. It abstracts without forgetting what the abstraction cost. This is the beginning of mathematical judgment. The conclusion returns to the book's central operation: recognizing sameness across difference. It asks whether mathematics is best understood not as a fixed body of eternal objects, but as the disciplined architecture through which difference becomes identity without becoming nothing.

Part III - The QRNG Experiments

Chapter 14: The First QRNG Pilot: Random Equations

Based on the archived v0.1.2 pilot report, preserved as a negative result.

The first QRNG experiment was intentionally more conservative. It did not generate mathematical objects directly. It froze a finite universe of short identities in one binary operation, used archived quantum bytes to rank that universe, and then examined a fixed sample. This was useful because it made the experimental chain easy to audit. It was also too close to conventional mathematical language.

The pilot selected an identity that appeared, on every three-element model satisfying it, to force two familiar-looking consequences: flexibility and left alternativity. If the story had stopped at carrier size three, the result could easily have been promoted as a theorem-like discovery. It was not. A four-element countermodel satisfied the selected axiom while violating both consequences. The experiment therefore produced a clean finite-model mirage.

That failure became the design lesson that made the later theorem possible. Small-model regularity is evidence, not proof. A QRNG-selected equation can be interesting without being new. And randomness spent on syntactic variants of the same law is wasted entropy. The next experiment needed to move below equations and into the generation of the object itself.

QRNG Equational Consequence Search v0.1.2 - Audit-Repair Pilot Report

Verdict

No universal theorem candidate survived.

The experiment did, however, expose a clean finite-model mirage and reject it with an explicit larger countermodel.

Provenance

According to the archived experimental chronology, the specification was frozen before the quantum draw. The package itself cannot independently establish that chronology because its local timestamp and hash files are mutable archival files rather than an external timestamp commitment. The first v0.1.0 specification was separately abandoned without randomization because the retrieval layer returned a cached response from the preceding experiment.

v0.1.1 instead precommitted to the distinct ANU request `length=10&type=uint8` and received

[250,170,143,46,53,106,51,247,61,230]

or seed hex

faaa8f2e356a33f73de6.

SHAKE-256 was used only to rank the frozen identity universe reproducibly.

Search outcome

The grammar contains four-leaf versus three-leaf identities in one binary operation. 128 identities were selected from the frozen universe.

Of those, 74 met the predeclared finite-model eligibility threshold.

The primary identity was selection-rank 46 / candidate 405:

(((zy)z)x) = y(z*x).

Renaming `(z,y,x)` to `(x,y,z)` gives the cleaner form

((xy)x)z = y(x*z).

It has 61 labeled three-element models, including 27 two-sided-nondegenerate models under the frozen definition.

Every three-element model satisfied two predeclared panel identities:

(xy)x = x(yx) [flexibility] (xx)y = x(xy) [left alternativity]

No other panel law was forced in all three-element models.

Universal proof gate

Neither candidate consequence could even begin a contextual rewrite proof from one side using the selected axiom: the relevant side contains no subterm matching either side of the axiom. The bounded proof search therefore returned no universal proof.

That alone means the experiment cannot claim a theorem.

Four-element countermodel

A direct countermodel search then found the operation table

- | 0 1 2 3

---+-------- 0 | 3 0 3 3 1 | 3 2 0 3 2 | 3 0 0 3 3 | 3 3 3 3

It satisfies

((xy)x)z = y(x*z)

for all 64 assignments on the four-element carrier.

But at x=1, y=2:

(xy)x = 0 x(yx) = 3,

so flexibility fails.

The same `(x,y)=(1,2)` also gives

(xx)y = 0 x(xy) = 3,

so left alternativity fails as well.

Thus the apparent three-element implications are definitively finite-size artifacts.

Meaning for the novelty experiment

This is a useful negative result because it distinguishes three layers that are easy to conflate:

- 1. a random axiom can have a striking finite-model regularity;
- 2. exhaustive verification at one carrier size is not a universal theorem;
- 3. an explicit equational proof or larger countermodel can decide which interpretation is warranted.

The current axiom is not promoted as novel mathematics.

v0.1.2 audit repair

The mathematical experiment and all committed experimental outputs are unchanged from v0.1.1. v0.1.2 strengthens release verification so that it now checks the entire archived chain from QRNG response to seed and then reruns the finite scan in a temporary directory. The regenerated `selected_identities.json`, `summary.json`, `proof_attempts.json`, and `candidate_theorem.json` must be byte-for-byte identical to the committed files.

The release wording now distinguishes internal pre-specification from independently timestamped preregistration. For a future experiment, the specification hash should be committed to an external timestamped system before requesting the random bytes.

A second design lesson is that the syntactic identity universe contains elementary symmetry duplication. That issue is intentionally not repaired retroactively; a future search should canonicalize variable-renaming and opposite-magma classes before random selection.

Chapter 15: The Object-First Turn

New synthesis of the research lineage and the v0.2.1 repository.

The object-first experiment changes the causal order of the research. The quantum-origin seed does not choose a theorem, a property, or a named algebraic class. It is deterministically expanded into labeled binary operations on a four-element carrier. Each table is then examined through transformations that the table itself induces: its left and right translations.

For an element a of a magma Q, define L_a(x)=ax and R_a(x)=xa. These are total functions on the carrier. Composing them produces a finite transformation monoid. Breadth-first search from the identity assigns to every generated transformation a shortest word length in the chosen translations. The largest such shortest length is the translation depth of the table.

This invariant was attractive experimentally for several reasons. It is exact on a finite carrier. It depends on the dynamics generated by the operation rather than on resemblance to a named algebraic species. It supports independent certification by ordinary breadth-first search. And, crucially, it can reveal an extremal phenomenon that was not encoded as an equation in the seed-generation stage.

From Quantum Randomness to a Theorem We Did Not Ask For

A research story behind QRNG Mathematical Abiogenesis

This repository did not begin with the theorem it now contains.

It began with a less precise question:

> *Can quantum randomness participate in the origination of mathematics rather than merely in computation, simulation, or randomized search?*

Random numbers have long been used to choose examples, initialize optimizers, and sample finite spaces. Replacing a pseudorandom source with a QRNG does not, by itself, make mathematics "quantum-born." The stronger goal is for the random event to occur before the mathematical target is known. The random source should generate an object, after which structure is measured, anomalies are recognized, conjectures are formulated, and exact mathematics is proved.

The present repository is an exploratory attempt to cross that boundary.

1. 2024: quantum search for mathematical truths

An early precursor was the 2024 paper Quantum Discovery: Exploring the Search for Mathematical Truths through Quantum Superposition.

That paper proposed generating populations of semi-random mathematical formulae, representing candidates in quantum superposition, and using quantum-oracle and amplitude-amplification ideas to favor valid equations. The ambition was already

recognizable: quantum mechanics would participate in mathematical discovery rather than merely speed up arithmetic.

The limitation is clearer in retrospect. The search space was still supplied in advance. Variables, operators, functions, grammar, validity criteria, and the very idea of searching formulae were human-defined. Quantum mechanics could help search the space, but it did not decide what mathematical object should exist.

The 2024 question was therefore:

> *Can quantum computation help search for new mathematics?*

The present project asks a harder question:

> *Can quantum randomness generate the object from which the mathematical question itself emerges?*

That shift-from searching statements to originating objects-became the central methodological turn.

2. 2025: randomness as substrate rather than noise

Two later lines of work changed the role assigned to QRNG output.

Structured Quantum Randomness

Structured Quantum Randomness: Filtering Quantum Number Generators and Double-Slit Interference as Computational Anchors treated quantum randomness not simply as high-quality noise but as a possible computational substrate. It asked whether structure could be extracted from quantum-generated bitstreams and whether apparently random output might reveal deeper organization under suitable transformations.

The current repository does not require the stronger claim that hidden structure exists inside QRNG output. Its theorem remains valid whether the seed is metaphysically irreducible, hidden-variable output, or simply an externally supplied finite byte string.

But one methodological lesson survived: a QRNG stream can be treated as a primitive input to a mathematical construction rather than as background noise.

Pure Information from Quantum Randomness

The 2025 paper Pure Information from Quantum Randomness: QRNGs as Interfaces Between Substrate, Qubits, and the It-From-Bit Cosmos pushed the idea differently. It treated QRNG events as unusually clean informational events and explicitly entertained quantum bits as seeds of novelty.

That paper was philosophical rather than theorem-generating. Yet it sharpened a useful operational question:

> *What happens if those bits are allowed to determine a mathematical world before we decide what we are looking for?*

3. 2026: mathematics before names

The decisive mathematical machinery came from the Algebraic Abiogenesis program.

Mathematical It from Bit

Mathematical It from Bit: Algebraic Abiogenesis, Anonymous Finite Operations, and the Automated Birth of Abstract Algebra reversed the conventional order of abstract algebra.

Instead of beginning with groups, rings, lattices, semigroups, or a supplied axiom system, the method begins with finite carriers and anonymous operation tables. Terms are evaluated exactly. Collisions of term functions reveal candidate identities. Degenerate structures are rejected. Countermodels eliminate finite coincidences. Surviving structure can then be compressed into laws, quotient objects, rewrite systems, and eventually recognizable algebra.

The central chain was:

```
finite anonymous operations
    -> exact observations
    -> invariants and identities
    -> countermodels
    -> theory
    -> names, if names are useful
```

This is much closer to the architecture needed for QRNG-born mathematics. It allows the random source to act below the level of named algebraic species.

Before Algebra Is Named

Before Algebra Is Named: Heidegger, Anonymous Operations, Mathematical Disclosure, and the Limits of Abiogenic Algebra added an important caution: anonymity is not absence of structure. An operation table may lack familiar labels while still being a completely specified mathematical rule.

This guards against a false claim of creation ex nihilo. The QRNG does not create mathematical truth by magic. It specifies a finite object. Mathematics begins when we investigate what follows from that object without having predetermined the theorem we hope to obtain.

Can Artificial Intelligence Discover Mathematics Before Humans Name It?

The 2026 review Can Artificial Intelligence Discover Mathematics Before Humans Name It? supplied a framework for distinguishing verification, proof discovery, conjecture discovery, concept invention, and theory or ontology formation. It also emphasized provenance, countermodels, reproducibility, novelty auditing, and the difference between solving a supplied problem and changing the specification of the problem itself.

The current QRNG experiment should be judged against those stronger criteria. It is not interesting merely because a computer enumerated 65,536 cases. The important question is where the theorem came from.

The Virgin Oracle

The Virgin Oracle: From a Finite Alphabet to Abstract Algebra explored a related thought experiment: how much mathematics could a nascent intelligence reconstruct from a deliberately impoverished symbolic starting point?

That work reinforced the idea that mathematical structure can be approached through finite primitives, repeated operations, equivalence, and invariant behavior rather than through inherited terminology. In the present project, the QRNG-origin finite operation plays the role of such a primitive oracle: not an oracle that announces truths, but one that supplies a world whose consequences must be learned.

4. The methodological synthesis

The weaker architecture is:

```
human defines candidate theorem language
    -> QRNG chooses a statement
    -> computer tests it
```

In that design, most mathematical content exists before the random event.

The stronger architecture is:

```
QRNG-origin bytes
    -> finite anonymous object
    -> intrinsic computation
    -> unexpected invariant
    -> conjecture not specified before the object existed
    -> exhaustive or deductive proof
    -> novelty audit
```

That is the architecture implemented here.

The experiment is still exploratory. The archived seed was inherited from an earlier QRNG experiment, and this repository does not independently authenticate the historical ANU acquisition. The future confirmatory protocol therefore requires a fresh 32-byte / 256-bit QRNG response obtained only after an externally timestamped specification is frozen.

But the mathematical chain in the executed run can be reconstructed exactly.

5. What the QRNG-origin object produced

The inherited 10-byte seed is deterministically expanded with SHAKE-256 into 250,000 labeled binary operations on the carrier `{0,1,2,3}`.

For a magma `Q`, every element `a` induces a left and right translation:

```
L_a(x) = a*x
R_a(x) = x*a
```

Those transformations generate a finite transformation monoid. The search measured the maximum shortest generator-word length needed to reach a transformation from the identity. In this repository it is called translation depth.

The first sampled table reaching depth 17 occurred at index 5849:

```
2 0 0 2
0 0 0 0
2 3 1 3
0 0 1 1
```

Its complete left/right translation monoid has 188 elements and depth 17.

The important anomaly appeared when only two translations were retained:

```
L_2 = (2,3,1,3)
R_3 = (2,0,3,1)
```

Those two maps alone generate a 176-element monoid of depth 17.

At this point the QRNG search had done its job. It had produced an object whose internal dynamics suggested a mathematical question that had not been used to select the seed:

> *Is 17 merely large for this sample, or is 17 the largest possible two-generator depth for total transformations on four points?*

The random experiment was then left behind.

6. The theorem no longer depends on randomness

There are exactly `4^4 = 256` total transformations of a four-element set and therefore `256^2 = 65,536` ordered pairs of generators.

The repository exhaustively enumerates all of them.

The result is exact:

> *Theorem. Among all ordered pairs of total transformations on a four-element set, the maximum generator-relative transformation-monoid diameter is 17.*

The enumeration further proves:

- exactly 144 ordered pairs attain depth 17;
- every extremal pair generates a 176-element monoid;
- the 144 pairs form 6 classes under simultaneous relabeling of the four points;
- after also identifying interchange of the two generators, they form 3 classes;
- no ordered pair has depth 16 or depth greater than 17.

The QRNG-origin magma contains one of those global extremizers.

This is the key epistemic separation in the project:

```
randomness discovered the question;
exhaustive mathematics established the answer.
```

The theorem would remain true if every historical claim about the seed were discarded tomorrow. Its proof is finite and independent of provenance. Provenance matters only for the stronger historical statement that a QRNG-origin object led us to formulate the theorem.

See THEOREM_REPORT.md for the exact statements and EXECUTED_PROTOCOL.md for the executed chronology.

7. Why this is different from the 2024 proposal

The path from the 2024 paper to this repository can be summarized as a sequence of removed assumptions.

2024

```
quantum process
    -> search semi-random formulae
    -> validate candidate truths
```

The mathematical language exists first.

Algebraic Abiogenesis

```
anonymous finite operation
    -> compute equivalences and invariants
    -> infer theory
```

The named theory no longer exists first.

QRNG Mathematical Abiogenesis

```
quantum-origin information
    -> anonymous finite operation
    -> unexpected dynamical invariant
    -> new mathematical question
    -> exact theorem
```

Now neither the particular object nor the final theorem is supplied in advance.

That does not prove autonomous mathematical creativity in any sweeping philosophical sense. It does instantiate a narrower, auditable proposition:

> *A physically generated random seed can determine an otherwise unspecified finite mathematical object, and analysis of that object can originate a rigorous theorem question that was not encoded as the search target.*

8. What we can and cannot presently claim

We can claim

The repository contains an exact, reproducible finite theorem and classification. The theorem was reached by an object-first exploratory search. The selected witness was generated from bytes archived as QRNG-origin data. The witness led to the two-generator extremal question. The global theorem was then proved by exhaustive enumeration independent of the random seed.

We cannot yet claim

We cannot prove from this repository alone that the inherited bytes were historically delivered by ANU. There is no independent external timestamp attesting to that acquisition.

We cannot claim that the exact theorem is historically first merely because the current novelty review did not locate an earlier statement. The repository therefore calls it a credible novelty candidate rather than a settled priority result.

We also cannot claim that 17 is the maximum full translation depth among all four-element magmas. The theorem concerns two total transformations on four points. Complete enumeration of all 4^16 binary operations was not performed.

Those limitations are part of the experiment.

9. The next experiment

The next confirmatory run should be cleaner than the historical path that produced this result.

Before acquiring any new quantum bytes:

- 1. freeze the object-construction algorithm;
- 2. freeze the invariant suite and promotion rules;
- 3. freeze the computational budget;
- 4. publish or externally timestamp the specification;
- 5. only then acquire 32 fresh QRNG bytes / 256 bits;
- 6. archive and timestamp the exact response immediately;
- 7. generate the mathematical object stream without human seed selection;
- 8. retain negative runs;
- 9. promote only anomalies satisfying predeclared criteria;
- 10. prove any resulting theorem independently of the random source;
- 11. conduct a specialist prior-art audit before making a priority claim.

That protocol separates three questions cleanly: randomness provenance, discovery provenance, and mathematical validity.

10. The larger research program

The long-term target is not to accumulate random curiosities.

The target is a repeatable architecture in which quantum-origin information enters below the level of mathematical vocabulary and occasionally forces the discovery system into regions humans did not choose in advance.

```
quantum event
    -> finite formal world
    -> intrinsic invariants
    -> anomaly
    -> new definition or conjecture
    -> proof
    -> classification
    -> literature comparison
    -> new mathematics, if it survives
```

The most interesting outcome would not be that a QRNG repeatedly finds isolated numerical records. It would be that one randomly generated world exposes a structural phenomenon rich enough to demand a new definition, an analytic explanation, and eventually a family of theorems.

The present degree-four theorem may or may not become such a family. Immediate questions include:

- Why is 17 the ceiling structurally, rather than merely computationally?
- Why do all extremizers generate monoids of size 176?
- Why are there exactly 6 simultaneous-conjugacy classes and 3 classes after generator interchange?
- Can the extremizers be characterized without exhaustive enumeration?
- What happens for five points?
- Can QRNG-origin finite algebras lead to invariants outside transformation depth entirely?

Those are now ordinary mathematical questions. That is exactly the point.

The QRNG was useful not because mathematics stayed random, but because randomness led us somewhere specific enough that randomness could be discarded.

Source lineage

• Quantum Discovery: Exploring the Search for Mathematical Truths through Quantum Superposition (2024).

• Structured Quantum Randomness: Filtering Quantum Number Generators and Double-Slit Interference as Computational Anchors (2025).

• Pure Information from Quantum Randomness: QRNGs as Interfaces Between Substrate, Qubits, and the It-From-Bit Cosmos (2025).

• Mathematical It from Bit: Algebraic Abiogenesis, Anonymous Finite Operations, and the Automated Birth of Abstract Algebra (2026).

• Before Algebra Is Named: Heidegger, Anonymous Operations, Mathematical Disclosure, and the Limits of Abiogenic Algebra (2026).

• Can Artificial Intelligence Discover Mathematics Before Humans Name It? Automated Theory Formation, Machine-Generated Mathematics, and Algebraic Abiogenesis (2026).

The executable culmination is this repository, especially THEOREM_REPORT.md, EXECUTED_PROTOCOL.md, NOVELTY_REVIEW.md, and EXPERIMENT_SPEC.md.

The story is intentionally historical and methodological. The theorem claims themselves are governed by the executable certificates and theorem report, not by this narrative.

Chapter 16: Translation Monoids and Depth

Based on the v0.2.1 README and executable definitions.

A generator-relative diameter should not be confused with the size of a monoid. Two generator pairs may produce the same number of transformations while requiring very different word lengths to reach the most distant element. The depth records how long the shortest necessary composition can become.

On a four-point set there are only 4^4=256 total transformations. An ordered pair (g,h) therefore lives in a finite universe of 256^2=65,536 possibilities. Starting from the identity map, one applies g and h, records any new transformations, and repeats. The first time a transformation appears fixes its shortest word length. When no new transformations appear, the search has generated the entire monoid <g,h>. The maximum recorded distance is diam(g,h).

The finiteness of this universe turns the later conjecture into an unusually clean theorem problem. Once a QRNG-origin magma supplied a pair with depth 17, there was no need to estimate the global maximum statistically. Every ordered pair could be checked. This is a recurring theme of the project: use open-ended generation to originate the question, then use closed, exact mathematics to answer it.

QRNG Mathematical Abiogenesis v0.2.1 - hardening release

This release preserves the mathematics of v0.2.0 and hardens the release boundary, provenance language, and prior-art review.

> *Can a finite object generated from a QRNG-origin seed lead, by object-first computation, to a rigorous theorem that was not supplied to the random generator in advance?*

Status

Exploratory theorem-bearing hardening release. The finite mathematics is exactly checkable. The exact `n=4,m=2` result remains a credible novelty candidate, not a settled priority claim.

No fresh post-freeze QRNG draw was executed. The search that produced the witness reused 10 bytes archived in earlier v0.1.2 work. This ZIP proves only the internal consistency of the archived response/seed artifacts; it contains no independent externally timestamped attestation of their historical ANU acquisition. See `EXECUTED_PROTOCOL.md` and `data/INHERITED_SEED_NOTICE.txt`.

The future confirmatory specification has been restored to 32 fresh bytes / 256 bits, with external timestamping required before candidate inspection. It is unexecuted in v0.2.1.

Object-first construction

The inherited seed deterministically expands via SHAKE-256 into 250,000 labeled binary operations on `{0,1,2,3}`. The domain separator remains the frozen v0.2.0 string so the executed stream is unchanged. For a magma `Q`, define

L_a(x) = ax R_a(x) = xa

and let `M(Q)` be the transformation monoid generated by all left and right translations. The translation depth `delta(Q)` is the largest shortest generator-word length of an element of `M(Q)`.

No conventional identity, variety, or named algebraic structure participates in witness selection.

Witness and theorem

The first sample record of depth 17 occurs at index 5849:

2 0 0 2 0 0 0 0 2 3 1 3 0 0 1 1

Its full left/right translation monoid has 188 elements and exact depth 17. Two translations already suffice:

L_2 = (2,3,1,3) R_3 = (2,0,3,1)

They generate a 176-element transformation monoid of depth 17. Exhaustive enumeration of all `256^2 = 65,536` ordered pairs proves:

> *Theorem. Among all ordered pairs of total transformations on a four-element set, the maximum generator-relative monoid diameter is 17.*

Furthermore, exactly 144 ordered pairs attain 17; all generate 176-element monoids; there are 6 extremal classes modulo simultaneous relabeling and 3 after also identifying generator interchange. The inherited-seed witness contains one of these global extremizers.

Additional exact results

- All `3^9 = 19,683` labeled three-element magmas: maximum full translation depth 8, attained by exactly 6 labeled tables.
- Fixed inherited-seed order-4 sample of 250,000 tables: sample maximum 17, first at index 5849; 29 sample tables attain 17.
- All 163,668 labeled tables within Hamming distance 1-4 of the witness: none exceeds the witness's computed base depth.
- The witness is nonassociative, noncommutative, nonidempotent, rigid, simple, and has no proper nonempty submagma; these were computed only after selection.

Verification

On a clean extraction:

python verify_manifest.py python verify_release.py

`verify_manifest.py` is fail-closed: any unlisted extra file causes failure. `verify_release.py` checks the complete file set/hashes, frozen specification hash, response/seed/provenance consistency, tests without cache/bytecode creation, the independent theorem verifier, and deterministic theorem-critical committed results.

The expensive exploratory replay remains explicit:

python scripts/reproduce.py python verify_release.py --full-replay

The full replay scans 250,000 order-4 tables and the complete radius-4 neighborhood and may be substantially slower than the quick release audit.

Claim discipline

Salomaa, Panteleev, Cameron-Castillo-Ramirez-Gadouleau-Mitchell, East-Egri-Nagy-Mitchell, and Ryzhikov establish close prior territory for function-composition depth, minimal word length, transformation-semigroup enumeration, and diameter. `NOVELTY_REVIEW.md` now treats these as required prior art. No source located in the review states the exact 17 / 144 / 176 / 6 / 3 result, but v0.2.1 does not infer priority from search absence.

Chapter 17: The Witness at Index 5,849

Based on EXECUTED_PROTOCOL.md and THEOREM_REPORT.md from v0.2.1.

The fixed exploratory stream contained 250,000 labeled order-four operation tables. The first table attaining the sample record depth 17 occurred at index 5,849. Its table is small enough to print in full, but its translation dynamics are unexpectedly deep.

The full translation monoid generated by all distinct left and right translations has 188 elements and depth 17. Post-selection analysis then revealed something stronger: two translations alone, L_2=(2,3,1,3) and R_3=(2,0,3,1), already generate a 176-element monoid of depth 17. This observation changed the mathematical problem. The question was no longer whether the sampled magma was unusually deep. It became whether any ordered pair of total transformations on four points could exceed depth 17.

That escalation was exploratory rather than preregistered. It is important to say so. The original frozen search did not contain the statement of the eventual two-generator theorem. The theorem question appeared only after inspection of the selected object. This is precisely why the episode is relevant to the larger theme of the book.

Executed exploratory protocol - inherited-seed run

This file describes the search that produced the witness and theorem carried into v0.2.1. The mathematical search stream is unchanged from v0.2.0.

Provenance status

No fresh post-freeze ANU response was obtained. The exploratory run reused the 10 archived bytes carried forward from `qrng-equational-consequence-search-v0.1.2`:

fa aa 8f 2e 35 6a 33 f7 3d e6

SHA-256:

a3597a1486262acab8a28873ec3c3a4358806821f32333944cb01862b7929762

The present ZIP verifies that `qrng_response.json`, the extracted binary seed, the hex rendering, and this digest are mutually consistent. It does not contain an independent externally timestamped attestation proving that the archived response was delivered by ANU at the historical acquisition time. Thus the theorem does not depend on the provenance claim, while the historical description "QRNG-origin" remains explicitly qualified.

Candidate stream

Carrier: `{0,1,2,3}`.

Frozen domain separator (retained verbatim for reproducibility despite the hardening release version):

qrng-mathematical-abiogenesis-v0.2.0/order4-table

For index `i`, blocks are

SHAKE256(domain || seed || uint64_be(i) || uint16_be(counter), 32 bytes).

Accept bytes `<252`, map accepted bytes modulo 4, and take the first 16 values as the row-major multiplication table. Sample indices are exactly `0..249999`.

Selection invariant

For each table `Q`, form all left and right translations and breadth-first search the generated transformation monoid from the identity. `delta(Q)` is the maximum shortest generator-word length. The primary witness is the lowest sample index attaining the sample maximum.

Post-selection theorem escalation

After the primary witness was fixed and the pair `L_2,R_3` was observed to retain depth 17, all `256^2` ordered pairs of total transformations on four points were exhaustively enumerated. This establishes the global two-generator theorem recorded in `THEOREM_REPORT.md`.

This theorem escalation was not part of a successfully executed pre-seed protocol. It is an exploratory conjecture-generation step followed by exhaustive proof.

Theorem report

v0.2.1 hardening note: the theorem statements and numerical classifications below are unchanged from v0.2.0. Only release integrity, provenance language, and review machinery were changed.

Definitions

Let `X={0,1,2,3}`. For total transformations `g,h : X -> X`, let `<g,h>` denote the transformation monoid generated by `g,h` and the identity under composition.

For `f in <g,h>`, define `ell_{g,h}(f)` as the length of a shortest word in `g,h` evaluating to `f`. Define

diam(g,h) = max_f ell_{g,h}(f).

This is generator-relative diameter/depth; it is not an intrinsic diameter of an abstract monoid independent of its generators.

Theorem 1 - exact total two-generator degree-4 maximum

max_{g,h in X^X} diam(g,h) = 17.

Certificate

There are `4^4=256` total transformations on X, hence exactly `256^2=65,536` ordered pairs. `results/two_generator_global.json` records the complete depth histogram after exhaustive BFS for every pair.

Exactly 144 ordered pairs have depth 17. No pair has depth 16 or greater than 17. Every depth-17 pair generates 176 transformations.

Under simultaneous conjugation by `S_4`, the 144 ordered extremizers form 6 classes. If generator interchange `(g,h)~(h,g)` is also allowed, they form 3 classes.

Theorem 2 - QRNG witness realizes the global pair maximum

The QRNG-derived order-4 magma at sample index 5849 is

- | 0 1 2 3

  --+-------- 0 | 2 0 0 2 1 | 0 0 0 0 2 | 2 3 1 3 3 | 0 0 1 1

  Its translations include

  L_2 = (2,3,1,3) R_3 = (2,0,3,1).

The pair `<L_2,R_3>` has 176 elements and depth 17, so it attains Theorem 1's global maximum.

The unique transformation at distance 17 is

(2,2,3,0).

One shortest word is

R3 R3 R3 L2 R3 R3 L2 R3 R3 R3 L2 R3 R3 L2 R3 R3 R3

with length 17.

The full left/right translation monoid has 188 elements and also has depth 17; the same transformation remains uniquely deepest.

Theorem 3 - exact order-3 magma baseline

Among all `3^9=19,683` labeled binary operations on `{0,1,2}`, the maximum full left/right translation depth is 8. Exactly 6 labeled operations attain 8. The complete histogram and all maximizers are in `results/order3_baseline.json`.

Theorem 4 - local robustness of the QRNG witness

Exactly 163,668 labeled order-4 tables differ from the witness in 1, 2, 3, or 4 cells. Exhaustive enumeration finds none with translation depth greater than 17. Therefore the witness is a radius-4 local maximizer in the labeled Hamming graph of operation tables.

This statement is label-dependent and is not a global order-4 optimum theorem.

What is not proved

This package does not prove that 17 is the maximum full left/right translation depth over all four-element magmas. There are `4^16` labeled order-4 magmas, and that complete enumeration was not performed.

Chapter 18: The Theorem We Did Not Ask For

The central exact result; theorem wording follows THEOREM_REPORT.md.

The exhaustive classification gives more than a maximum. Exactly 144 ordered pairs attain depth 17. Every one of them generates a monoid of size 176. Under simultaneous relabeling of the four points, those 144 pairs collapse to six classes. If generator interchange is also identified, the six collapse to three.

The repeated monoid size 176 is not a consequence of the definition of depth; it is an additional regularity of the extremizers. Likewise, the absence of depth-16 pairs is a feature of the complete enumeration rather than a sampling artifact. These facts are mathematically interesting because they suggest a structural explanation still waiting to be extracted from the computation.

The theorem is therefore not the end of the research program. A computational proof answers what the maximum is. A structural proof would explain why the ceiling is 17, why the extremal monoids have size 176, and why the symmetry classes occur in the observed counts. The QRNG-origin experiment has converted an alien finite table into ordinary mathematical questions that no longer depend on the QRNG.

Theorem report

v0.2.1 hardening note: the theorem statements and numerical classifications below are unchanged from v0.2.0. Only release integrity, provenance language, and review machinery were changed.

Definitions

Let `X={0,1,2,3}`. For total transformations `g,h : X -> X`, let `<g,h>` denote the transformation monoid generated by `g,h` and the identity under composition.

For `f in <g,h>`, define `ell_{g,h}(f)` as the length of a shortest word in `g,h` evaluating to `f`. Define

diam(g,h) = max_f ell_{g,h}(f).

This is generator-relative diameter/depth; it is not an intrinsic diameter of an abstract monoid independent of its generators.

Theorem 1 - exact total two-generator degree-4 maximum

max_{g,h in X^X} diam(g,h) = 17.

Certificate

There are `4^4=256` total transformations on X, hence exactly `256^2=65,536` ordered pairs. `results/two_generator_global.json` records the complete depth histogram after exhaustive BFS for every pair.

Exactly 144 ordered pairs have depth 17. No pair has depth 16 or greater than 17. Every depth-17 pair generates 176 transformations.

Under simultaneous conjugation by `S_4`, the 144 ordered extremizers form 6 classes. If generator interchange `(g,h)~(h,g)` is also allowed, they form 3 classes.

Theorem 2 - QRNG witness realizes the global pair maximum

The QRNG-derived order-4 magma at sample index 5849 is

- | 0 1 2 3

  --+-------- 0 | 2 0 0 2 1 | 0 0 0 0 2 | 2 3 1 3 3 | 0 0 1 1

  Its translations include

  L_2 = (2,3,1,3) R_3 = (2,0,3,1).

  The pair `<L_2,R_3>` has 176 elements and depth 17, so it attains Theorem 1's global maximum.

  The unique transformation at distance 17 is

  (2,2,3,0).

  One shortest word is

  R3 R3 R3 L2 R3 R3 L2 R3 R3 R3 L2 R3 R3 L2 R3 R3 R3

  with length 17.

The full left/right translation monoid has 188 elements and also has depth 17; the same transformation remains uniquely deepest.

Theorem 3 - exact order-3 magma baseline

Among all `3^9=19,683` labeled binary operations on `{0,1,2}`, the maximum full left/right translation depth is 8. Exactly 6 labeled operations attain 8. The complete histogram and all maximizers are in `results/order3_baseline.json`.

Theorem 4 - local robustness of the QRNG witness

Exactly 163,668 labeled order-4 tables differ from the witness in 1, 2, 3, or 4 cells. Exhaustive enumeration finds none with translation depth greater than 17. Therefore the witness is a radius-4 local maximizer in the labeled Hamming graph of operation tables.

This statement is label-dependent and is not a global order-4 optimum theorem.

What is not proved

This package does not prove that 17 is the maximum full left/right translation depth over all four-element magmas. There are `4^16` labeled order-4 magmas, and that complete enumeration was not performed.

Part IV - What We Can Claim

Chapter 19: Discovery Provenance, Novelty, and the Firewall

Based on NOVELTY_REVIEW.md and the v0.2.1 claim-discipline notes.

Three different claims must be kept separate.

The first is mathematical validity. The finite theorem is established by exhaustive enumeration and can be reproduced without any QRNG data. The second is discovery provenance: the research chronology records that the witness was produced from bytes archived as originating in an ANU QRNG response. The current standalone package checks the internal consistency of the response, seed, hexadecimal rendering, and digest, but it does not contain an independently timestamped external attestation proving the historical acquisition event. The third is historical priority: a literature search has not located the exact 17/144/176/6/3 result, but absence from the searched literature is not proof that no one previously computed it.

This separation is not bureaucratic caution. It prevents three very different kinds of evidence from being substituted for one another. A theorem does not become false if its discovery story is uncertain. A genuine QRNG provenance record would not make a false theorem true. And a correct theorem discovered through an interesting process is not automatically historically novel. The repository deliberately treats these as independent gates.

Novelty review - strengthened, still provisional

Search date: 2026-09-02.

Claim under review

The package does not claim novelty for left/right translations, multiplication/translation monoids, shortest generator words, transformation depth, or semigroup diameter. The narrow candidate is:

> *For total transformations on a four-element set, the maximum generator-relative diameter over ordered two-generator systems is 17; 144 ordered pairs attain it, all generating 176-element monoids; the extremizers form 6 simultaneous-conjugacy classes and 3 classes after generator interchange.*

Directly relevant prior art

- 1. A. Salomaa, Composition sequences for functions over a finite domain, Theoretical Computer Science 292(1) (2003), 263-281. DOI: `10.1016/S0304-3975(01)00227-4`. Salomaa explicitly develops depth and complete depth for compositions of functions on a finite domain. This is essential terminology/metric prior art.
- 2. P. Panteleev, Preset Distinguishing Sequences and Diameter of Transformation Semigroups, LATA 2015, pp. 353-364; arXiv:1412.0034. DOI: `10.1007/978-3-319-15579-1_27`. Panteleev studies maximal subsemigroup diameter in the full transformation semigroup and its asymptotics.

- 3. P. J. Cameron, A. Castillo-Ramirez, M. Gadouleau, J. D. Mitchell, Lengths of words in transformation semigroups generated by digraphs, Journal of Algebraic Combinatorics 45 (2017), 149-170. DOI: `10.1007/s10801-016-0703-9`. This directly studies minimal generator-word lengths for transformations in generated transformation semigroups.
- 4. A. Ryzhikov, Careful synchronisation and the diameter of transformation semigroups with few generators, arXiv:2506.13962 (2025). This is the closest current framing found: transformation depth / automaton diameter and `max-diam(n,m)` for few generators.
- 5. J. East, A. Egri-Nagy, J. D. Mitchell, Enumerating transformation semigroups, Semigroup Forum 95 (2017), 109-125. DOI: `10.1007/s00233-017-9869-2`. It enumerates transformation semigroups through degree 4 up to several equivalences. This is particularly important because the present exact `n=4` result may be derivable from pre-existing enumeration data even if not stated as a diameter theorem.
- 6. M. Beaudry, Finite idempotent groupoids and regular languages, RAIRO Theoretical Informatics and Applications 32 (1998), 127-140. DOI: `10.1051/ita/1998324-601271`. This uses the multiplication monoid of finite groupoids, establishing nearby binary-algebra territory.
- 7. A. Drápal, T. Kepka, P. Maršálek, Multiplication Groups of Quasigroups and Loops II, Acta Universitatis Carolinae Mathematica et Physica 35 (1994). Left/right translations and multiplication groups are standard in the bijective/quasigroup setting.

Searches performed

Targeted searches included variants of:

- `max-diam(4,2)` and `max diam 4 2 transformation semigroup`;
- `degree 4 transformation semigroup diameter 17`;
- `4 states two generators transformation depth`;
- `composition sequences finite domain depth`;
- `minimal word length transformation semigroup`;
- `multiplication monoid magma diameter`;
- exact small-order multiplication/translation-monoid classifications.

The review also checked the degree-4 enumeration literature as the most plausible place for an unstated duplicate.

Result of the review

No located source states the exact theorem `max-diam_total(4,2)=17`, the count of 144 ordered extremizers, the common monoid size 176, or the 6/3 extremal-class counts.

That absence is not proof of priority. The exact value may exist in specialist databases, code, theses, unpublished computations, or be immediately extractable from existing enumerations. Consequently v0.2.1 uses only credible novelty candidate / provisional novel-mathematics candidate, never "first proof," "previously unknown," or an unqualified priority claim.

QRNG-discovery claim discipline

The discovery path in the executed exploratory run was:

archived seed claimed to originate from ANU QRNG -> finite binary operation -> translation-depth outlier -> observation that L2,R3 alone retain depth 17 -> conjecture that 17 is extremal for two total transformations on 4 points -> exhaustive proof over all 65,536 ordered pairs.

The theorem's truth is independent of the seed provenance. This standalone package proves only internal consistency of the inherited seed artifacts; it does not independently authenticate their historical ANU origin.

Chapter 20: The Confirmatory Experiment

Based on the frozen future confirmatory specification in v0.2.1. This protocol was not executed in the present release.

The next experiment should remove the principal historical ambiguity. Before any fresh random bytes are acquired, the complete object-generation algorithm, invariant suite, promotion rule, computational budget, and stopping conditions should be frozen and externally timestamped. Only then should a 32-byte, 256-bit QRNG response be obtained and archived with its own external timestamp.

The larger seed is not mathematically necessary for unbiased randomization, but it improves the credibility of the provenance story by making retrospective seed grinding implausible. Negative runs should be retained. The experiment should not be allowed to substitute a more attractive witness after the fact. Any conjecture promoted from a generated object should then be proved by a method that no longer depends on the random source.

The goal is not to ritualize preregistration. It is to make the origin of the mathematical question auditable. If a future run produces nothing interesting, that negative result should remain part of the record. If it produces an anomaly, the question is whether that anomaly can be converted into a theorem, definition, or classification that survives ordinary mathematical scrutiny.

QRNG Mathematical Abiogenesis v0.2.1 - Future Confirmatory Specification

Purpose

Use a post-freeze quantum random seed to generate finite binary algebras directly, then extract a rigorous mathematical statement from the intrinsic dynamics of those generated objects.

This release does not let the QRNG choose among a prewritten list of equations. The QRNG seed generates multiplication tables. The primary invariant is then computed from those tables without attaching a conventional algebraic name to them first.

The experiment may establish an explicit existence/lower-bound theorem. It must not claim a global order-4 optimum unless that optimum is independently proved.

Carrier and binary structures

The carrier is the labeled set

$A = \{0,1,2,3\}$.

A candidate is a total binary operation `* : A x A -> A`, stored as its 16 row-major table entries.

No associativity, identity, cancellation, commutativity, idempotence, or other algebraic law is imposed.

Quantum seed

After this specification is frozen, SHA-256 hashed, and externally timestamped, request 32 fresh `uint8` values (256 bits) from the Australian National University QRNG service and archive the exact JSON response. Immediately create an external timestamp/archive record for that exact response before examining any generated candidate.

Thirty-two bytes are intentionally retained even though far fewer bits are sufficient for unbiased randomization: the larger seed makes retrospective seed grinding materially less plausible and gives a cleaner confirmatory provenance record.

The 4 returned bytes are the only claimed quantum entropy. SHAKE-256 below is a deterministic expander and contributes no new entropy.

Candidate stream

The fixed stream length is 250,000 candidate tables, indexed `i = 0,...,249999`.

Use domain separator

qrng-mathematical-abiogenesis-v0.2.0/order4-table

For candidate index `i`, generate bytes in blocks

SHAKE256(domain || seed || uint64_be(i) || uint16_be(counter), 32 bytes)

for counters `0,1,2,...` until 16 table entries have been obtained.

For exact unbiased conversion from bytes to elements of A:

- accept a byte only when it is < 252;
- map an accepted byte to `byte mod 4`;
- append accepted values in returned order;
- stop after 16 values.

Because 252 is divisible by 4, this introduces no modulo bias.

Left/right translations

For a candidate magma Q and each `a in A`, define transformations of A

L_a(x) = a x R_a(x) = x a.

Let

G(Q) = {L_0,L_1,L_2,L_3,R_0,R_1,R_2,R_3}

with duplicate transformations removed without changing first-occurrence order.

Let `M(Q)` be the transformation monoid generated by `G(Q)` together with the identity map under ordinary function composition.

Translation depth

For `f in M(Q)`, let `ell_Q(f)` be the minimum number of generators from `G(Q)` whose composition equals f. The empty word represents the identity and has length zero.

Define the translation depth

delta(Q) = max_{f in M(Q)} ell_Q(f).

Compute `M(Q)`, all shortest distances, and a shortest word for every generated transformation by breadth-first search from the identity. The BFS itself is the exact certificate of `|M(Q)|` and `delta(Q)`.

Primary QRNG witness

Evaluate all 250,000 candidates. Let

d_sample = max_i delta(Q_i).

The primary witness is the smallest candidate index i attaining `d_sample`.

Record every strict depth record encountered in stream order.

No alternative witness may be substituted because it has a larger monoid, prettier table, more familiar properties, or a stronger literature narrative.

Theorem gate

The package may state the following theorem after independent verification:

> *There exists a four-element magma Q whose left/right translation monoid has translation depth d_sample.*

Equivalently, if

D_4^LR = max{delta(Q) : Q is a binary operation on a four-element set},

then

D_4^LR >= d_sample.

The package must not replace >= by = without a separate complete global proof.

Local robustness gate

For the primary labeled table Q, exhaustively enumerate every labeled four-element table differing from Q in 1, 2, 3, or 4 of its 16 entries. Each changed entry may take any of its other three values.

There are exactly

sum_{k=1}^4 C(16,k) 3^k = 163,668

such neighbors.

Record the maximum translation depth among these neighbors and its distribution by Hamming radius. A statement that Q is a radius-4 local maximizer is allowed only if no neighbor has depth greater than `delta(Q)`.

This neighborhood result is label-dependent and must be described as such.

Exact order-3 baseline

Independently enumerate all `3^9 = 19,683` labeled binary operations on `{0,1,2}`. Compute the corresponding left/right translation depth exactly and report the global maximum and count of labeled maximizers.

This provides a complete small-order baseline. It does not imply an order-4 optimum.

Conventional-property firewall

Only after the primary witness is fixed may the package compute descriptive properties such as associativity, commutativity, idempotence, automorphism count, congruences, or named identities. None of these properties participates in witness selection.

Novelty firewall

After the theorem is fixed:

- 1. search literature using `multiplication semigroup`, `multiplication monoid`, `translation semigroup`, `translation monoid`, `left and right translations`, transformation-semigroup word length, and generator/Cayley diameter terminology;
- 2. distinguish the package's generator-relative depth from other notions called semigroup diameter;
- 3. search specifically for exact small-order classifications of multiplication/translation monoids of arbitrary magmas/groupoids;
- 4. do not call the result novel merely because the QRNG-generated table is new;
- 5. state novelty only at the level supported by the literature search.

Reproducibility

Archive:

- frozen specification and SHA-256;
- freeze timestamp record;
- exact QRNG response and extracted 32-byte seed;
- seed SHA-256;
- strict-record sequence;
- primary table;
- full shortest-distance and shortest-word certificate for its translation monoid;
- order-3 exhaustive baseline;
- radius-4 neighborhood summary;
- all code, tests, and release manifest.

A confirmatory release verifier must check the frozen-specification hash, external timestamp records, exact QRNG response/seed consistency, complete fail-closed manifest, tests, independent theorem verification, and deterministic result files. The

expensive 250,000-candidate and radius-4 replays may be exposed as an explicit full-replay mode rather than silently hidden inside a quick verifier.

Chapter 21: Evaluation, Certification, and Reproducibility

Adapted from Can Artificial Intelligence Discover Mathematics Before Humans Name It? (2026), Sections 11-12.

A theorem-discovery system is only as strong as the gates that prevent rediscovery, finite coincidence, hidden human targeting, and unverifiable provenance from being promoted as novelty.

The success of Algebraic Abiogenesis cannot be measured merely by the number of identities generated, the volume of computation completed, or the unfamiliarity of its outputs. A system can produce enormous quantities of formally correct but mathematically unimportant material. It can also rediscover known laws under obscure notation and mistake unfamiliar presentation for genuine novelty. Evaluation must therefore address several distinct questions. Did the machine generate the result without being given the intended target? Is the result

mathematically correct? Does it survive changes in carrier size, operation population, and search policy? Is it deductively nonredundant? Does it compress or explain a meaningful class of structures? Is it historically new? Can another investigator reproduce the entire discovery process? These questions require different forms of evidence. Exact evaluation certifies that a collision holds in a particular finite algebra. Countermodels define the limits of generalization. Automated proof establishes implication from an identified law basis. Formal proof assistants certify logical derivations. Literature comparison addresses historical priority. Repeated experiments test robustness. Public repositories preserve provenance and permit independent execution.

No single certificate is sufficient. A formally verified identity may be trivial. A novel finite pattern may not generalize. A robust law may already be classical. A productive theory may depend strongly on a hidden human choice in the search grammar. The evaluation of machinegenerated mathematics must therefore be layered, adversarial, and transparent.

11.1 What Counts as a Machine-Discovered Mathematical Theory

A mathematical theory is more than a set of true equations. It contains a population of objects or models, a vocabulary for describing them, a collection of governing laws, and relationships among those laws. It distinguishes primitive assumptions from derived consequences and organizes examples, counterexamples, constructions, and transformations into a coherent structure. For a machine-generated output to count as a discovered theory, several conditions should be met. First, the output must exhibit organization. The system should identify a stable core of laws, not merely report isolated collisions. It should determine which identities are fundamental, which are redundant, and which distinguish neighboring classes of operations.

Second, the output must define or isolate a nontrivial model population. A proposed theory should classify more than one accidental operation table unless the single

structure itself has exceptional significance. The theory should identify a reproducible family of operations or explain why a particular finite algebra is structurally distinguished. Third, the theory should possess deductive productivity. Its central laws should generate additional consequences through proof, rewriting, or closure. A law set that supports no meaningful deductions beyond the observations from which it arose remains closer to a data summary than to a mathematical theory.

Fourth, the theory should remain stable under reasonable perturbations of the experiment. Small changes in enumeration order, term representatives, sampling seeds, or implementation details should not destroy its central structure. Fifth, the output should support interpretation. This need not mean immediate human naming. It means that the system can explain how the theory separates models, compresses behavior, or generates consequences. Machine discovery does not require complete autonomy. Human investigators may choose the broad domain, approve evaluation criteria, and judge significance. The relevant standard is whether the decisive law structure was explicitly supplied or emerged through the computational process.

11.2 Rediscovery, Novelty, and Historical Comparison

Rediscovery and novelty should be treated as different achievements. Rediscovery is valuable because it validates the discovery mechanism. If a system beginning with anonymous operations independently recovers associativity, commutativity, idempotence, absorption, mediality, or other recognized laws, then the process has demonstrated that familiar algebraic structure can emerge from exact behavior without being named in advance. Such recovery is strongest when the system also reconstructs relationships among the laws. It should not merely detect associativity in one table. It should recognize a family of associative operations, identify consequences of associativity, and distinguish associative from nearby nonassociative branches.

Historical novelty requires a higher evidential burden. A law may be absent from a local database but well known under another notation. Two equations may look different while defining equivalent varieties. A theory may duplicate an established class described in another branch of universal algebra. Novelty assessment should therefore compare candidate results at several levels: literal equation search, variable renaming, duality, term substitution, definitional equivalence, interpretability, and deductive equivalence. Automated literature search can assist, but expert review remains necessary because mathematical concepts often migrate across terminology and disciplines.

The system should preserve three labels rather than force a premature binary classification: known, apparently novel, and unresolved. An apparently novel result becomes a defensible discovery only after sustained comparison and independent scrutiny.

Rediscovery should not be concealed or treated as failure. A transparent system should report exactly which outputs recover known theories and which resist identification. The mixture itself provides evidence about the search process.

11.3 Syntactic Novelty Versus Semantic Novelty

Syntactic novelty is easy to generate. A machine can produce an unfamiliar equation by increasing term depth, permuting variables, or composing known laws into elaborate forms. Such an expression may be absent from the literature while containing no new mathematics. Semantic novelty concerns the behavior or theory defined by the expression. Two syntactically different identities may have the same consequences. One may be derivable immediately from another. A complicated law may define a familiar variety under a disguised presentation. The evaluation process must therefore quotient candidate identities by appropriate notions of equivalence. At the simplest level, equations should be normalized under variable renaming, symmetry, and reversal. More deeply, automated provers should test mutual implication between candidate laws or law sets.

If theory T_1 proves every axiom of T_2 and T_2 proves every axiom of T_1, the two presentations are deductively equivalent even if their surface forms differ. Such equivalence does not make the alternative basis worthless. A shorter, more elegant, or computationally effective basis may still be a contribution. But the novelty claim should concern the presentation rather than the underlying theory. Semantic novelty may also arise from a new model class, a new boundary between known classes, a previously unknown finite basis, or an unexpected relationship among established theories. The discovery need not introduce an entirely unprecedented algebraic universe to be significant.

This distinction protects the project from one of the easiest forms of self-deception: confusing symbolic unfamiliarity with mathematical invention.

11.4 Compression, Explanatory Power, and Theorem Productivity

Compression is one of the strongest available measures of theory quality. A useful law replaces many separate observations with one generative principle. A compact basis may explain thousands of term-function collisions as consequences of a small number of identities. Compression can be measured at several levels. Syntactic compression compares the size of the retained axiom set with the raw identity catalog. Semantic compression measures how many operation tables or term behaviors are classified by the theory. Computational compression

measures whether the theory permits faster prediction of term values or equivalence than exhaustive evaluation. Compression alone is not enough. A constant operation may generate extreme collapse among terms, yet the resulting theory may be mathematically simple. A stronger theory may compress more because it has eliminated almost all variation. Explanatory power requires that the retained laws reveal why the observed regularities occur together. If associativity explains a large family of parenthesization collisions, it provides more than a compact record. It identifies the structural source of those collisions. Theorem productivity offers another criterion. After a theory basis is proposed, the system can measure how many nontrivial consequences follow, how reusable its lemmas become, and whether new constructions or classifications emerge. A fertile theory should support continued research rather than terminate in a static summary.

These measures must be normalized carefully. A theory that proves everything because its assumptions are inconsistent has maximal apparent productivity and no mathematical value. Consistency checks and model witnesses are therefore essential.

11.5 Robustness Across Finite Operations and Carrier Sizes

A law discovered in one finite algebra may reflect a genuine structural principle or an accident of small size. Robustness testing distinguishes these possibilities. The system should evaluate candidate identities across diverse operation populations. These may include exhaustive collections at small carrier sizes, random operations, operations evolved to challenge the current theory, and representatives selected to maximize structural diversity. Persistence across carrier sizes is particularly important. An identity that appears on carriers of size two and three but disappears at size four should not be promoted as a general law without qualification. Conversely, a theory that recurs independently across several sizes gains credibility.

Robustness should also be tested against changes in search policy. The core results should not depend entirely on one variable set, term-depth cutoff, enumeration order, or random seed. Sensitivity analysis should identify which conclusions remain invariant and which are artifacts of the observational protocol. This does not imply that size-specific theories are uninteresting. Finite algebra contains genuine phenomena restricted to particular cardinalities. The requirement is accurate scope. The system should distinguish a universal candidate from a law of a bounded model population.

A robust result is one whose domain of validity is clearly characterized and reproducibly detected.

11.6 Countermodel Resistance and Generalization

Countermodel search is the principal defense against premature generalization. Every proposed implication should be subjected to an adversarial search for a structure satisfying the premises and violating the conclusion. The search should begin with small carriers and expand as resources permit. Constraint solvers and finite model generators can target candidate counterexamples more efficiently than undirected random sampling. Survival against bounded countermodel search does not prove universal validity. It strengthens the conjecture and establishes a verified lower bound on the size of any counterexample. Proof remains necessary for unrestricted generality.

Countermodels should also guide theory repair. If a candidate law fails, the system should compare the countermodel with positive examples and search for the smallest additional condition that restores the pattern. The result may be a refined theorem or a division of the model population into separate branches. Generalization should proceed through alternating induction and criticism:

```
observed collision → candidate law → countermodel search → theory refinement → deductive
```

proof. This cycle is stronger than frequency-based learning because each failure changes the theory's architecture. The machine does not merely lower confidence in a statement. It acquires a new structural witness.

11.7 Formal Certification and Proof-Assistant Integration

Formal certification should be reserved for results that have survived earlier filters. It would be wasteful to formalize every raw collision. The strongest candidates should first be normalized, compared for redundancy, tested across models, and organized into provisional theories. A proof assistant can then certify several kinds of claims: that an operation table satisfies a stated identity, that one law follows from others, that an axiom basis is sufficient for a theorem, or that a computational procedure correctly enumerates a bounded space. Formalization does not by itself certify novelty, importance, or intended meaning. It establishes correctness relative to the encoded definitions and assumptions. Semantic review remains necessary to ensure that the formal statement corresponds to the mathematical claim described in prose.

The ideal architecture separates trusted and untrusted components. AI agents may generate definitions, proofs, and code. The final derivations should be checked by a small trusted kernel. Computational certificates should be independently replayable. Operation tables and countermodels should be included as explicit data rather than summarized only in narrative form. Formal certification is especially valuable when theories are unfamiliar. Human intuition provides less protection against subtle error in a newly generated ontology. Exact checking becomes a substitute for familiarity.

11.8 GitHub Repositories as Executable Scientific Archives

A conventional paper records conclusions and selected methods. An executable repository can preserve the entire discovery process. For Algebraic Abiogenesis, the repository should contain the term grammar, operation generators, exact evaluation code, canonicalization procedures, raw or compressed signatures, candidate identities, countermodels, proof artifacts, configuration files, and scripts needed to regenerate the principal results. Version history records how the theory changed. Failed branches and rejected conjectures can be preserved without occupying the main paper. Issues can document unresolved problems. Releases can define stable experimental milestones.

The repository therefore functions as more than supplementary material. It becomes part of the scientific claim. A reader should be able to determine whether the reported theory truly emerged from the stated inputs and policies. Reproducibility also requires environmental control. Software versions, dependency locks, random seeds, carrier enumerations, and computational limits should be recorded. Expected outputs should be accompanied by checksums or machine-verifiable summaries. An executable archive also permits continuation. Other investigators can extend carrier sizes, modify grammars, introduce additional operations, or replace ranking policies. The research becomes a platform rather than a closed report.

This continuation function is especially important for machine-generated mathematics. The repository preserves not only what was discovered but how further discovery can proceed.

11.9 Human Oversight, Provenance, and Intellectual Accountability

Human oversight remains necessary at every stage where mathematical meaning, historical novelty, or scientific significance is judged.

The human investigator should not be required to perform every linear computation. That would defeat one purpose of AI-assisted research. Oversight should concentrate on the policies governing the search, the interpretation of results, and the claims made publicly. Provenance must identify which components were human-specified and which were machinegenerated. This includes initial grammars, operation populations, ranking criteria, code, conjectures, proofs, and narrative interpretation. Such disclosure makes the division of labor inspectable. Intellectual accountability cannot be delegated to an AI system. The named authors remain responsible for verifying that evidence supports the conclusions, that comparisons with prior work are fair, and that uncertainty is stated honestly.

The most dangerous failure is not a visible computational error. It is an inflated conceptual claim built on technically correct output. A system may find a real identity and still fail to discover a theory. It may form a coherent theory and still fail to establish novelty. It may produce a novel finite phenomenon and still fail to justify broad philosophical conclusions about autonomous mathematics. Human governance should therefore impose graduated claims. The language of observation, conjecture, theorem, theory, and novelty should be used distinctly. Each level should require its own certificate. The result is not diminished by caution. On the contrary, a transparent hierarchy of evidence makes machine-generated mathematics credible. Algebraic Abiogenesis will be judged not only by what it discovers but by whether it establishes a rigorous standard for showing that a mathematical theory has genuinely emerged.

12. Toward an Autonomous Ecology of Mathematical Discovery

The long-term significance of artificial intelligence for mathematics will not be determined solely by whether machines solve harder benchmark problems. The deeper transformation will occur if computational systems begin to sustain mathematical research programs: generating questions, inventing concepts, testing conjectures, preserving counterexamples, comparing theories, and revising their own representations over extended periods. Such systems would not merely produce answers. They would maintain ecologies of possible mathematics. Algebraic Abiogenesis provides a restricted experimental setting in which this possibility can be studied. Anonymous finite operations offer exact semantic environments. Generated terms provide a controlled expressive language. Term-function collisions supply candidate laws. Proof search, countermodels, and theory comparison convert those observations into evolving

structures. Human judgment governs the direction and significance of the work, while AI performs much of the sequential labor. The objective is not to construct a conscious artificial mathematician or transfer a human mind into a machine. A keyboard and monitor provide sufficient separation. The machine may remain an external instrument while assuming increasingly sophisticated responsibilities. What matters is whether the combined system can preserve conceptual continuity, execute long investigations, and produce mathematical organization that neither participant would have produced alone.

The frontier is therefore not machine autonomy in the absolute sense. It is governed autonomy inside a transparent research ecology.

12.1 From Theorem Proving to Ontology Formation

The progression from calculation to theorem proving, conjecture generation, concept invention, and theory formation represents an increasing transfer of mathematical initiative to machines. A theorem prover operates inside an ontology. The objects, predicates, operations, axioms, and desired conclusion are already present. Its task is to discover a valid path through that world. Ontology formation occurs when the world itself becomes part of the search. The system must determine which distinctions deserve preservation, which equivalences define stable objects, which operations are fundamental, and which laws organize the resulting domain. This transition changes the nature of evaluation. A theorem prover can be judged by correctness, proof length, or success rate. An ontology-forming system must also be judged by coherence, compression, fertility, robustness, and explanatory power. It must produce not only valid propositions but useful mathematical worlds.

Algebraic Abiogenesis approaches ontology formation by withholding conventional interpretations from finite operations. The system does not begin by declaring that an operation is multiplication or conjunction. It allows term behavior to determine which equations become visible and which operation populations belong together. The resulting ontology is still constrained. The system begins with finite sets, total operations, variables, and a term grammar. It therefore does not discover the very idea of algebra from an unrestricted universe. It explores how much algebraic organization can emerge once a thin operational substrate has been supplied.

This limitation is scientifically useful. Ontology formation should be studied in stages rather than treated as an all-or-nothing mystery. A controlled experiment can reveal which structures

emerge under particular observational policies and which require additional representational resources. The next stage may allow the system to revise its own signature. Repeated behavior could motivate the introduction of constants, unary operations, derived relations, partial operations, or multiple interacting operations. At that point, the language of the theory would no longer remain completely fixed during discovery. A still more advanced system could compare alternative ontologies. One representation might organize the data through equations, another through order, another through transformations or categories. The system would then search not only for laws inside a language but for the language that best exposes the structure.

12.2 Human Nonlinear Insight and AI Linear Execution

Human and machine intelligence need not imitate one another to collaborate effectively. Their differences can be treated as complementary resources. Human mathematical insight is often nonlinear. A researcher may move abruptly between domains, notice an analogy without being able to justify it immediately, or sense that a technically correct approach is conceptually wrong. Ideas may emerge through accumulated experience rather than through a visible chain of deductions. Such thinking can appear scattered when judged by the standards of programming or formal proof. Yet the apparent disorder may be the mechanism by which distant structures are brought into contact. The polymath's value lies partly in the ability to maintain several incomplete conceptual worlds and recognize an unexpected bridge among them.

Programming demands a different mode. Definitions must be explicit. Interfaces must be fixed. Edge cases must be handled. Each step must be translated into a sequence executable by a machine. This discipline is necessary for reproducibility, but prolonged immersion in it can consume the attention required for broad conceptual synthesis. AI permits a new division of labor. The human can propose the leap. The machine can construct the bridge. In practice, the human may state a broad objective, recognize a failed ontology, or reject a result as trivial. The AI can translate that intervention into code changes, experiments, exhaustive searches, proof obligations, documentation, and repository updates.

This division is not a rejection of rigor. It can increase rigor by allowing the human to remain at the appropriate level of abstraction while the machine performs exact linear work. The danger

is not that the machine handles implementation. The danger is that the human accepts the machine's implementation without adequate semantic inspection. The research loop must therefore alternate between modes:

```
human conceptual intervention → AI operationalization → exact computation → AI organization
→ human judgment.
```

Neither participant alone defines the complete process. The human without computational execution may remain at the level of speculation. The machine without conceptual governance may optimize sterile objectives or generate vast quantities of trivial structure.

12.3 The Preservation of Separation Between Mathematician and Machine

The increasing power of AI does not require psychological, biological, or legal fusion between person and machine. Separation can be a design principle. The keyboard and monitor create a clear boundary. The human issues instructions, observes results, rejects errors, and chooses which branches to continue. The machine remains external, inspectable, and interruptible. This boundary preserves responsibility. The researcher cannot attribute a false public claim to an autonomous system while still accepting credit for successful results. The human author remains accountable for the interpretation and publication of machine-assisted work. Separation also protects intellectual independence. An AI system may propose persuasive but misleading formulations. The physical and conceptual distance of an external interface reminds the user that machine output is an artifact for evaluation rather than an internal intuition.

The alternative vision of consciousness transfer is unnecessary for mathematical collaboration. A person's memories, preferences, or research methods need not be uploaded to gain the advantages of machine execution. Structured instructions, repositories, notebooks, and recorded decisions can provide sufficient continuity. The aim is not to reproduce the mathematician inside software. It is to externalize selected cognitive burdens. This distinction also clarifies what can persist after the original investigator stops directing the work. A repository may preserve a discovery method, evaluation policy, and open problem list. Future humans and machines can continue the program without claiming that the originating person has survived as a conscious entity.

The preserved object is a research lineage, not necessarily a person.

12.4 Open-Ended Theory Populations and Evolutionary Search

Open-ended mathematical discovery may be better represented as a population of theories than as a single path toward one optimum. Each candidate theory can be defined by a model population, a law basis, a deductive closure, and a record of counterexamples. Theories may share common cores while differing through additional identities. Some may merge when their apparent differences prove superficial. Others may divide when new countermodels expose hidden heterogeneity. Evolutionary search offers one mechanism for exploring this landscape. Candidate theories can vary through the addition, removal, substitution, or recombination of identities. Their model populations can be regenerated or challenged. Their productivity can be measured through the consequences and constructions they enable.

Selection need not produce one winner. Different theories may survive because they optimize different criteria. One may offer the shortest basis. Another may classify the broadest model family. Another may support unusually rich derived operations. A fourth may connect unexpectedly to a known domain. Diversity is therefore not merely computational waste. It protects the system from premature convergence. A single global score might eliminate a theory that later becomes important under a different interpretation. An ecological architecture should preserve rare but promising branches while allocating most resources to productive regions. It should record why branches were pruned and permit their restoration if later discoveries change the evaluation.

Theory evolution must also remain mathematically disciplined. Random mutation of equations can generate endless novelty without significance. Variation should be connected to observed countermodels, proof dependencies, or structural gaps. A new law should enter because it explains behavior, repairs a failure, or opens a productive branch. The resulting process resembles neither ordinary theorem proving nor unrestricted symbolic generation. It is a controlled evolution of mathematical organizations under exact criticism.

12.5 Machine-Learned Mathematical Interestingness

Interestingness is the central unsolved problem of open-ended mathematical discovery. Once machines can generate more candidate mathematics than humans can inspect, selection becomes the primary bottleneck.

Fixed measures capture only fragments of value. Short identities are easy to understand but may be trivial. Rare identities may be novel but arbitrary. Strong theories may collapse model diversity. Productive theories may generate many unimportant consequences. Machine-learned interestingness would infer patterns from the history of mathematical development and from the internal success of the discovery process. The system could learn which concepts later become widely reused, which conjectures create new branches, and which definitions compress previously disconnected results. Such learning must be handled cautiously. Historical mathematics reflects human institutions, fashions, and available technologies. A model trained to imitate historical attention may reproduce disciplinary bias and overlook forms of mathematics that humans have not valued.

Internal measures can provide a partial alternative. A candidate theory may gain value when it:

- explains many independent collisions;
- supports short proofs of previously separate facts;
- persists across diverse models;
- produces useful derived constructions;
- separates natural operation populations;
- connects distant branches of the theory ecology;
- remains stable under changes in search policy.

Human feedback can supplement these measures. The mathematician may mark a result as elegant, trivial, surprising, or conceptually important. Over time, the system can learn the investigator's standards without treating them as universal truth. The evaluation policy should remain explicit and revisable. A theory selected under one notion of interestingness should be distinguishable from a theory forced by the raw data. This separation prevents aesthetic or strategic preferences from being disguised as mathematical necessity.

12.6 Cross-Domain Extension Beyond Finite Algebra

Finite algebra is an ideal initial domain because it permits exhaustive evaluation and exact certification. The broader vision, however, extends beyond anonymous binary operations. Logical systems provide one direction. Instead of beginning with known connectives, a machine could explore anonymous truth-functional operations and discover stable calculi, equivalence

classes, or circuit structures. NAND-based investigations are especially relevant because complex logical behavior can emerge from one primitive operation. Combinatorics provides another direction. Anonymous incidence relations, transformations, or local assembly rules could generate candidate invariants and classification theories before familiar objects are named. Dynamical systems could be approached through finite transition rules. The machine could discover equivalence classes of trajectories, conserved patterns, or emergent coarse-grained objects. Category-theoretic environments present a more difficult extension. Rather than beginning with elements and operation tables, the system could begin with anonymous objects and composable arrows, searching for stable universal patterns, equivalences, and factorization structures.

Scientific applications introduce approximation and noise. Here exact term-function collision must be replaced or supplemented by statistical equivalence, error bounds, and intervention tests. The risk of false regularity becomes substantially greater. Cross-domain extension should therefore proceed gradually. The finite algebraic setting can establish the architecture's basic principles: target-free generation, exact comparison, countermodel criticism, theory branching, and delayed interpretation. Each additional domain requires a clear account of what replaces exact finite semantics. The goal is not to impose algebraic methods everywhere. It is to determine whether the deeper discovery pattern generalizes:

```
minimally interpreted substrate → generated observations → stable equivalences → candidate
laws → theory ecology → interpretation.
```

12.7 Risks of Triviality, Rediscovery, and Hidden Human Targets

The most immediate risk is triviality. Finite operations often produce many term collisions, especially when the carrier is small or the operation is degenerate. A system may generate impressive volumes of output while discovering only constant behavior, projections, or obvious consequences. A second risk is rediscovery disguised by notation. The machine may produce a familiar identity in a complicated form and present it as new. Deductive comparison and historical search are necessary before novelty is claimed. A third risk is hidden targeting. Researchers may say that the system is open-ended while choosing grammars, model populations, and scoring functions designed to recover a desired

structure. The target may be absent from the prompt but embedded in the experimental design. A fourth risk is parameter dependence. A theory may emerge only under one depth cutoff, carrier size, sampling method, or ranking policy. Without sensitivity analysis, accidental artifacts may appear robust. A fifth risk is anthropomorphic interpretation. The use of words such as discovery, creativity,

understanding, and theory can imply mental capacities not demonstrated by the system. Operational definitions should replace rhetorical inflation. A sixth risk is computational opacity. An agentic system may generate code and results faster than humans can inspect them. Formal checking can certify some claims, but implementation errors, data leakage, and semantic misalignment remain possible.

A seventh risk is publication abundance without consolidation. AI can produce papers, repositories, conjectures, and formal artifacts at extraordinary speed. Without synthesis, the scientific record may become harder rather than easier to understand. These risks should be treated as research problems. A credible system should include triviality detectors, known-theory matching, target-audit procedures, parameter perturbation, independent reruns, and explicit uncertainty labels. The standard should not be whether the machine produces something unfamiliar. It should be whether the process makes a mathematically disciplined distinction between accident, rediscovery, refinement, and genuine novelty.

12.8 A Research Agenda for Post-Axiomatic Mathematical Discovery

Post-axiomatic discovery does not reject axioms. It investigates how candidate axioms can emerge before they are declared fundamental. A practical research agenda begins with replication. The operation-first pipeline should be executed across independently generated finite operation populations and carrier sizes. The same stable theory branches should recur under transparent conditions. The next requirement is deductive compression. Large collision sets should be reduced to small bases, with explicit proofs that the remaining identities follow. Countermodel search should then establish boundaries and independence. Each proposed implication should be challenged, and minimal separating structures should be preserved.

Historical comparison must determine which branches correspond to known varieties and which remain unidentified. This comparison should include alternative presentations and not rely on literal equation matching alone. The system should then measure theorem productivity. Each theory basis can seed selfsupervised theorem generation, derived-operation search, construction discovery, and classification experiments. A higher-order theory space should be built in which branches are related by implication, interpretation, refinement, duality, and shared model populations. The research program should also test grammar revision. Can recurrent behavior motivate the introduction of constants, secondary operations, or quotient objects without direct human specification?

Formal certification should accompany the strongest claims. Proof assistants can verify implication chains, computational enumeration procedures, and bounded completeness results. Finally, the system should be evaluated by independent teams. Reproduction by the original developers is necessary but insufficient. Outside investigators should be able to regenerate the results, modify the policies, and determine whether the reported theory ecology persists. This agenda converts a philosophical ambition into a sequence of falsifiable experiments.

12.9 Can Mathematics Be Born Before It Is Named?

Mathematics cannot arise from literal nothing. Every discovery process begins with distinctions, representations, and permissible operations. Even a human mathematician inherits language, notation, examples, and standards of proof. The meaningful question is whether names and axioms must come first. Algebraic Abiogenesis proposes that behavior can precede declaration. Anonymous operations can be examined before they are classified. Terms can be generated before laws are selected. Exact collisions can reveal regularities before a theory has been named. Countermodels can divide the emerging world before its taxonomy is known. If stable law systems then organize model populations, support deductions, survive perturbation, and resist historical reduction, mathematics has emerged in a significant operational sense. The system has not created structure from nothing. It has converted minimally interpreted behavior into a new level of formal organization.

The strongest result would be a theory whose discovery path is fully reproducible but whose final structure was not anticipated by the human investigators. It would possess explicit models,

a compact law basis, nontrivial consequences, clear boundaries, and a defensible relation to prior mathematics. Such an achievement would not eliminate human mathematics. Human judgment would remain necessary to recognize meaning, compare histories, choose future directions, and decide whether the theory deserves attention. The machine would have supplied something different: the sustained linear process through which a mathematical world became visible. The answer to the title question must therefore remain conditional. Mathematics may be born before it is named if birth is understood as the emergence of stable, generative, and certifiable structure from behavior. Naming follows when humans recognize the structure, interpret its significance, and place it within the larger history of mathematical thought.

Chapter 22: Toward QRNG-Born Mathematics

The strongest version of QRNG-born mathematics would not consist of a long list of random curiosities. It would produce a structural phenomenon rich enough to require new language. A quantum-origin seed would determine a formal world; invariant analysis would reveal a pattern; the pattern would demand a definition; the definition would support a theorem family; and the resulting mathematics would become independent of the event that first pointed toward it.

The present result is smaller. A four-element magma led to an extremal pair of transformations and an exact finite theorem. Yet its smallness is scientifically useful. Every step can be inspected. The negative pilot is preserved. The witness is explicit. The enumeration is complete. The provenance limitation is stated. The novelty claim is bounded. This gives us a laboratory in which the phrase "mathematics from quantum randomness" can be made operational rather than rhetorical.

Perhaps the most important reversal is the last one. Randomness is not valuable here because it makes mathematics random. It is valuable because it can push the search into a place we did not choose, after which mathematics becomes exact again.

Appendix A: Exact Theorem Summary

v0.2.1 hardening note: the theorem statements and numerical classifications below are unchanged from v0.2.0. Only release integrity, provenance language, and review machinery were changed.

Definitions

Let `X={0,1,2,3}`. For total transformations `g,h : X -> X`, let `<g,h>` denote the transformation monoid generated by `g,h` and the identity under composition.

For `f in <g,h>`, define `ell_{g,h}(f)` as the length of a shortest word in `g,h` evaluating to `f`. Define

diam(g,h) = max_f ell_{g,h}(f).

This is generator-relative diameter/depth; it is not an intrinsic diameter of an abstract monoid independent of its generators.

Theorem 1 - exact total two-generator degree-4 maximum

max_{g,h in X^X} diam(g,h) = 17.

Certificate

There are `4^4=256` total transformations on X, hence exactly `256^2=65,536` ordered pairs. `results/two_generator_global.json` records the complete depth histogram after exhaustive BFS for every pair.

Exactly 144 ordered pairs have depth 17. No pair has depth 16 or greater than 17. Every depth-17 pair generates 176 transformations.

Under simultaneous conjugation by `S_4`, the 144 ordered extremizers form 6 classes. If generator interchange `(g,h)~(h,g)` is also allowed, they form 3 classes.

Theorem 2 - QRNG witness realizes the global pair maximum

The QRNG-derived order-4 magma at sample index 5849 is

- | 0 1 2 3

  --+-------- 0 | 2 0 0 2 1 | 0 0 0 0 2 | 2 3 1 3 3 | 0 0 1 1

Its translations include

L_2 = (2,3,1,3) R_3 = (2,0,3,1).

The pair `<L_2,R_3>` has 176 elements and depth 17, so it attains Theorem 1's global maximum.

The unique transformation at distance 17 is

(2,2,3,0).

One shortest word is

R3 R3 R3 L2 R3 R3 L2 R3 R3 R3 L2 R3 R3 L2 R3 R3 R3

with length 17.

The full left/right translation monoid has 188 elements and also has depth 17; the same transformation remains uniquely deepest.

Theorem 3 - exact order-3 magma baseline

Among all `3^9=19,683` labeled binary operations on `{0,1,2}`, the maximum full left/right translation depth is 8. Exactly 6 labeled operations attain 8. The complete histogram and all maximizers are in `results/order3_baseline.json`.

Theorem 4 - local robustness of the QRNG witness

Exactly 163,668 labeled order-4 tables differ from the witness in 1, 2, 3, or 4 cells. Exhaustive enumeration finds none with translation depth greater than 17. Therefore the witness is a radius-4 local maximizer in the labeled Hamming graph of operation tables.

This statement is label-dependent and is not a global order-4 optimum theorem.

What is not proved

This package does not prove that 17 is the maximum full left/right translation depth over all four-element magmas. There are `4^16` labeled order-4 magmas, and that complete enumeration was not performed.

Appendix B: Verification Guide

The repository is designed so that theorem validity can be checked independently of the QRNG narrative. On a clean extraction, the quick audit verifies the fail-closed manifest, specification hash, response/seed consistency, tests, independent theorem verifier, and theorem-critical deterministic outputs.

```
python verify_manifest.py
python verify_release.py
```

The expensive replay remains explicit rather than hidden inside the quick verifier:

```
python scripts/reproduce.py
python verify_release.py --full-replay
```

A meaningful independent audit should also reimplement the 65,536-pair enumeration using a different representation, because agreement between two implementations is stronger evidence than rerunning the same code path.

References and Source Lineage

Douglas C. Youvan. Quantum Discovery: Exploring the Search for Mathematical Truths through Quantum Superposition. 2024.

Douglas C. Youvan. Structured Quantum Randomness: Filtering Quantum Number Generators and Double-Slit Interference as Computational Anchors. 2025.

Douglas C. Youvan. Pure Information from Quantum Randomness: QRNGs as Interfaces Between Substrate, Qubits, and the It-From-Bit Cosmos. 2025.

Douglas C. Youvan. Mathematical It from Bit: Algebraic Abiogenesis, Anonymous Finite Operations, and the Automated Birth of Abstract Algebra. 2026.

Douglas C. Youvan. Can Artificial Intelligence Discover Mathematics Before Humans Name It? Automated Theory Formation, Machine-Generated Mathematics, and Algebraic Abiogenesis. 2026.

Douglas C. Youvan. Before Algebra Is Named: Heidegger, Anonymous Operations, Mathematical Disclosure, and the Limits of Abiogenic Algebra. 2026.

Douglas C. Youvan. The Virgin Oracle: From a Finite Alphabet to Abstract Algebra. 2026.

A. Salomaa. Composition sequences for functions over a finite domain. Theoretical Computer Science 292(1), 263-281 (2003). DOI 10.1016/S0304-3975(01)00227-4.

P. Panteleev. Preset Distinguishing Sequences and Diameter of Transformation Semigroups. LATA 2015, 353-364; arXiv:1412.0034.

P. J. Cameron, A. Castillo-Ramirez, M. Gadouleau, J. D. Mitchell. Lengths of words in transformation semigroups generated by digraphs. Journal of Algebraic Combinatorics 45, 149-170 (2017).

J. East, A. Egri-Nagy, J. D. Mitchell. Enumerating transformation semigroups. Semigroup Forum 95, 109-125 (2017).

A. Ryzhikov. Careful synchronisation and the diameter of transformation semigroups with few generators. arXiv:2506.13962 (2025).

M. Beaudry. Finite idempotent groupoids and regular languages. RAIRO Theoretical Informatics and Applications 32, 127-140 (1998).