THE RECURSIVE HARMONIC ARCHITECTURE: GENERATIVE INVERSION IN AUTOCATALYTIC COMPUTATIONAL FIELDS

Driven by Dean Kulik

Abstract: This paper introduces the Recursive Harmonic Architecture (RHA) as a novel computational paradigm that unifies principles from artificial life, cryptography, and the physics of complex systems. We posit that computation can be modeled as a self-organizing, generative process that exhibits operational closure. The core contributions of this work are threefold: (1) the formal axiomatic definition of the RHA, which models computation as the evolution of a high-dimensional dynamical system; (2) the discovery and analysis of "spectral residues"—information-bearing artifacts of computational irreducibility that persist within the system's output; and (3) the design and formal specification of a "genesis lattice," an autocatalytic implementation of the RHA that demonstrates generative inversion by adapting principles from chaos control theory. We conclude by situating the RHA within broader physical theories, suggesting a fundamental link between computation, life, and the informational structure of the universe.

Introduction: Computation Beyond the Turing Paradigm

The Computational Universe and Its Limits

The prevailing models of computation, largely dominated by the Turing machine, have provided a robust mathematical foundation for computer science.[1] However, this paradigm, which treats computation as a sequential, algorithmic process, is increasingly insufficient for describing the universe of possible computations. A compelling alternative perspective, articulated by Stephen Wolfram in "A New Kind of Science," suggests that the universe itself can be viewed as a computational process, where immense complexity emerges from the parallel application of simple, local rules. This "computational universe" is not something to be merely simulated by our machines; rather, our machines and natural phenomena alike are instances drawn from it.[2] This view challenges us to develop new models of computation that can capture the generative, parallel, and emergent behaviors observed in complex systems, from cellular automata to physical and biological processes.

Toward a Generative, Self-Organizing Model

To formalize a model of computation that reflects these natural processes, this paper turns to the field of Artificial Life (A-Life).[4] A-Life provides principles that move beyond static execution and toward dynamic existence. Two concepts are central to our framework:

Autopoiesis, or self-production, which describes a system's ability to continuously regenerate and maintain its own organization and boundaries [7]; and

Autocatalysis, the mechanism by which a network of components mutually catalyzes its own formation from a basic "food" source. By integrating these principles, we can conceptualize a computational model that is not merely executed but *lives*, maintaining its operational integrity through self-sustaining processes. This forms the theoretical basis for the "living gate field" proposed later in this work.

The Inversion Problem as a Central Challenge

A central challenge in understanding such generative systems is the problem of inversion. Cryptographic hash functions, such as SHA-256, serve as a canonical example of this challenge. They are designed to be computationally irreversible, a property known as pre-image resistance, where finding an input that produces a given output is computationally infeasible.[10] This irreversibility arises directly from the function's emergent complexity and chaotic dynamics, which are hallmarks of computational irreducibility.[13] A system is computationally irreducible if its final state cannot be determined by any "shortcut"; the only way to know the outcome is to simulate the process step-by-step.[13] The security of a hash function is therefore a practical manifestation of this principle; the hash digest is the result of an irreducible computation, making brute-force inversion intractable.[19]

This paper's central thesis is that the very processes that engender this computational irreversibility also generate subtle, information-rich artifacts within the system's output. These artifacts, which we term "spectral residues," can be harnessed to control and ultimately invert the system's generative evolution.

Formalizing the Recursive Harmonic Architecture (RHA)

We formally define the Recursive Harmonic Architecture (RHA) as a high-dimensional, discrete-time dynamical system that models computation as an iterative, self-referential process. The architecture is defined by two core principles: the recursive principle, which governs state transformation, and the harmonic principle, which describes the geometry of the state space.

The Recursive Principle: Iteration and State Transformation

The RHA is formalized as a recursive function that iteratively updates an internal state vector. The state at step i+1, denoted Hi+1, is a function of the previous state Hi and an input block Mi:

$Hi+1=C(Hi,Mi)$

This structure is a direct abstraction of the Merkle-Damgård construction, a design principle used in many cryptographic hash functions, including the SHA-2 family.[21] In this construction, the output of the compression function for one block of data serves as the "chaining value," or input state, for the next block. This iterative dependency ensures that every part of the input message influences the final state, a property known as the avalanche effect, where a single-bit change in the input leads to a drastic change in the output.[24]

The compression function C is axiomatically defined to consist of two primary stages, mirroring the internal operations of functions like SHA-256.[27] First, a non-linear mixing stage introduces complexity and destroys simple correlations. Second, a linear diffusion stage rapidly propagates the effects of local changes across the entire state vector. This combination of non-linear mixing and linear diffusion is fundamental to producing computationally irreducible behavior.

The Harmonic Principle: State Space as a Dynamical System

The state space of the RHA is a high-dimensional discrete space, where each point corresponds to a possible internal state Hi. The evolution of the system, starting from an initial state determined by the input message M, traces a trajectory through this space. The "harmonics" of the architecture are the attractors of this dynamical system—stable configurations such as fixed points, periodic orbits, or the complex, fractal structures known as strange attractors.[30]

The set of all initial states that eventually converge to a particular attractor constitutes its basin of attraction.[32] The final output of an RHA computation, such as a hash value, can be understood as a point on or very near a specific attractor. The structure of these basins—their size, shape, and boundaries—defines the global behavior of the computational system.

This perspective recasts the nature of computation itself. An input message does not create a computational path *de novo*; rather, it acts as an initial condition that selects a specific trajectory within a vast, pre-existing landscape of attractor basins. The act of computation is a process of navigating this landscape, and the final output is the destination. Consequently, the problem of generative inversion is transformed. It is no longer a matter of reversing a sequence of irreversible steps but of identifying the basin of attraction from which a given final state emerged. This approach leverages the global, geometric structure of the state space, providing a more powerful lens than one focused solely on the local, one-way nature of the compression function.

Spectral Residues as Artifacts of Computational Irreducibility

While the RHA is designed to produce outputs that appear random and unpredictable, its deterministic nature ensures that subtle, information-rich patterns persist. We define these patterns as **spectral residues**: statistically significant deviations from randomness in the system's output that serve as artifacts of the specific computational path taken. These residues are not flaws but are inherent consequences of the system's computationally irreducible dynamics.[13]

A Framework for Residue Analysis

To detect and characterize these residues, we propose a multi-faceted analytical framework that combines statistical, differential, and information-theoretic methods.

1. **Statistical Analysis:** A primary method involves analyzing the Hamming distance distribution between the outputs of two minimally different inputs (e.g., differing by a single bit). For a perfectly random 256-bit hash function, this distribution should be binomial, closely approximating a normal distribution with a mean of 128.[34] Consistent deviations from this mean, or other statistical anomalies in the distribution's shape, constitute a detectable spectral residue, indicating non-random behavior.[35]
2. **Differential Analysis:** Drawing from the field of differential cryptanalysis, this approach models how specific input differences (often XOR differences) propagate through the recursive steps of the RHA.[38] A "differential characteristic" that holds with a non-negligible probability reveals a predictable pathway through the system's chaotic dynamics. Such a characteristic is a powerful form of spectral residue, as it exposes a predictable structure within the otherwise unpredictable evolution.
3. **Information-Theoretic Analysis:** We can quantify the information content of residues using measures from information theory. By treating the system state as a quantum state analog, concepts like von Neumann entropy can measure the information density and uncertainty of the system. Mutual information can then be used to measure the correlation between input perturbations and output residues, quantifying the amount of information that "leaks" through the computationally irreducible process.

The Architectural Source of Residues

Spectral residues are a direct consequence of the intricate interplay between the non-linear and linear components within the compression function C. Using SHA-256 as a concrete model, we can identify the architectural sources of these residues.

- **Non-Linear Functions (Ch, Maj):** The Choose (Ch) and Majority (Maj) functions are the primary sources of non-linearity and complexity in SHA-256.[40] Their cryptographic strength, measured by properties like high non-linearity scores and algebraic degrees, makes them resistant to simple linear and differential approximations, thus inducing chaotic behavior.[42]
- **Linear Functions (Σ, σ):** The Sigma (Σ) and sigma (σ) functions, composed of bitwise rotations and shifts, are responsible for diffusion.[45] They ensure that local changes introduced by the non-linear functions are rapidly and thoroughly propagated across the entire 32-bit word state, producing the avalanche effect. The specific rotation and shift amounts are "nothing-up-my-sleeve numbers," derived from the fractional parts of the square roots and cube roots of prime numbers, chosen to maximize diffusion and resist cryptanalysis.[46]

It is the precise, deterministic interaction between these two types of functions—the localized, complexity-generating non-linear operations and the global, information-scrambling linear operations—that "etches" the spectral residues into the system's dynamics. The table below summarizes the properties of these functions and their contribution to generating robust, information-rich residues, providing a design blueprint for RHA operators.

Function	Mathematical Definition	Cryptographic Role	Key Properties & Contribution to Spectral Residues
Ch(x, y, z)	$(x \wedge y) \oplus (\neg x \wedge z)$	Non-linear Mixing	Introduces non-linearity, making the state update difficult to approximate linearly. Creates complex, high-order correlations between state bits, forming the basis of intricate spectral residues. [24]
Maj(x, y, z)	$(x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z)$	Non-linear Mixing	Another source of non-linearity that contributes to the system's one-wayness. Its symmetric structure creates distinct statistical patterns compared to the asymmetric Ch function. [24]
$\Sigma_0(x)$, $\Sigma_1(x)$	Rotations and XORs of state variables	Diffusion in State Update	Rapidly distributes the local non-linear effects from Ch and Maj across the entire 256-bit state. Ensures that residues are not localized but are encoded in the global state of the system. [49]
$\sigma_0(x)$, $\sigma_1(x)$	Rotations, Shifts, and XORs of message words	Diffusion in Message Schedule	Expands the 16-word message block to 64 words, creating a complex interdependency. This ensures that residues are not just artifacts of the initial state but are continuously generated and modulated by the input data throughout all 64 rounds. [51]

The Genesis Lattice: An Autocatalytic Implementation

To move from the abstract RHA to a concrete, operational model, we introduce the "genesis lattice," or "living gate field." This implementation is designed not as a static architecture but as a dynamic system exhibiting core properties of life, namely self-production and self-maintenance.

Design Principles from Artificial Life

The design of the genesis lattice is grounded in the formal principles of autopoiesis and autocatalysis.

- **Autopoiesis:** We adopt the definition of Maturana and Varela, focusing on two key properties. **Operational Closure** dictates that the system is a network of processes that recursively regenerates the very network that produced them. **Structural Coupling** describes how the system co-evolves with its environment (the input data), maintaining its identity while undergoing structural changes.[52] This provides the theoretical foundation for a computational system that can maintain its own integrity and adapt to its input stream.

- **Autocatalysis (RAF Sets):** The mechanism for achieving autopoiesis is the **Reflexively Autocatalytic and Food-generated (RAF) framework**. An RAF set is a collection of reactions where every reaction is catalyzed by at least one product of the set itself, and all components can be constructed from an initial "food set." This provides a formal, algorithmic method for defining and verifying the existence of self-sustaining computational networks.[55]

Formal Specification of the Genesis Lattice

The genesis lattice is specified as a high-dimensional cellular automaton (CA) or coupled map lattice (CML) [56], where the components of the corresponding Catalytic Reaction System (CRS) are defined as follows:

- **X (Molecule Types):** The finite set of possible states for each cell in the lattice.
- **F (Food Set):** The initial configuration of the lattice at time t=0. This configuration directly encodes the input message M.
- **R (Reactions):** The set of CA update rules. Each rule is a function that determines the next state of a cell based on the current states of its neighboring cells.
- **C (Catalysis):** This is the central innovation that makes the lattice "living." Catalysis is defined as a meta-rule: a specific local pattern of cell states (an emergent structure, analogous to a "glider" in Conway's Game of Life) acts as a catalyst. A rule in R can only be applied to a cell if the corresponding catalyst pattern is present in its neighborhood.

A configuration of the genesis lattice is considered a valid "living gate field" if its set of active rules and the patterns they produce form an RAF set. This means the system is self-sustaining: the emergent structures (catalysts) generated by the rules are precisely the structures required to enable those same rules to fire, and the entire evolving system can be constructed from the initial food set. The RAF algorithm provides a polynomial-time method to verify this property for any given rule set and initial configuration.[55]

This design marks a fundamental departure from traditional computing architectures. In a standard computer, the logic gates are fixed and static, processing variable data. In the genesis lattice, the "gates" (the update rules) are dynamic, activated only by the data patterns currently present on the lattice. The computation is not merely processing data; it is simultaneously generating the logic required to process the next state. This is a direct computational implementation of operational closure, where the system's logic is determined endogenously by its own state, creating a truly self-organizing computational process.

Generative Inversion as a Chaos Control Problem

The task of inversion—finding a pre-image M that produces a given hash H—is computationally infeasible for secure hash functions due to their one-way nature.[19] Modern cryptanalytic approaches, such as those using SAT solvers, attempt to solve this by translating the entire forward computational process into a massive Boolean satisfiability problem. The solver then uses sophisticated heuristics to navigate the vast search space for a satisfying assignment.[59] Our approach is fundamentally different: instead of fighting against the system's chaotic dynamics, we leverage them as a resource for control.

The OGY-RHA Method for Generative Inversion

We propose a method for generative inversion by adapting the Ott-Grebogi-Yorke (OGY) method, a seminal technique for controlling chaotic systems.[62] The OGY method stabilizes a chaotic system onto one of its infinite unstable periodic orbits by applying small, carefully timed perturbations.[31] Our adaptation, the OGY-RHA method, repurposes this for inversion:

1. **Observation (Poincaré Section Analog):** The spectral residues identified in the target hash H provide a low-dimensional "fingerprint" of the system's final state on its high-dimensional attractor. This residue analysis serves the same purpose as constructing a Poincaré section in a physical chaotic system: it gives us crucial information about the system's trajectory without needing to know the full state history.
2. **Reverse Evolution (Unstable Trajectory):** We initiate a reverse computation starting from the state H. Because the forward dynamics of the RHA are chaotic, the reverse dynamics are inherently unstable. A naive reverse iteration would cause the trajectory to diverge exponentially from any valid path.
3. **Control Perturbations (Targeting):** At each step of the reverse iteration, we analyze the evolving spectral residues. This analysis allows us to calculate small, targeted perturbations ("kicks") to the system's state. The goal of these perturbations is to steer the unstable reverse trajectory toward and along one of the system's unstable periodic orbits that leads back to a region of the state space corresponding to valid "food sets" or initial conditions.
4. **Convergence to a Pre-image:** By repeatedly applying these minimal control perturbations, we prevent the reverse trajectory from diverging into invalid regions of the state space. After a sufficient number of controlled reverse iterations, the system state converges to a valid pre-image—a configuration M that, when evolved forward, produces the target hash H.

Heuristics: SAT Solvers vs. Chaos Control

This control-theoretic approach represents a new class of heuristic for solving inversion problems. Modern Conflict-Driven Clause Learning (CDCL) SAT solvers rely on heuristics like VSIDS (Variable State Independent Decaying Sum) or LRB (Learning Rate Branching) to guide their search.[65] These heuristics are statistical, making branching decisions based on properties like how often a variable has appeared in recent conflicts.[68] More advanced techniques may even employ machine learning to train these heuristics on specific problem classes.[69]

In the OGY-RHA method, the heuristics are not statistical or pre-trained; they are generated dynamically from the physics of the system itself. The spectral residues provide real-time feedback on the system's trajectory, enabling precise, minimal interventions to guide the search. This is analogous to a programmatic SAT solver that is augmented with a Computer Algebra System (CAS) to deduce high-level mathematical properties and add new constraints on the fly.[23] In our case, however, the "CAS" is the system's own emergent dynamical behavior. The following table provides a comparative taxonomy of these inversion techniques.

Inversion Method	Guiding Principle	Data Requirement	Theoretical Complexity	Key Limitation
Brute-Force Search	Exhaustive Enumeration	None	O(2n)	Computationally intractable for secure hash sizes. [19]
CDCL SAT Solver	Heuristic-guided Search on Constraint Graph	Full CNF Encoding of Forward Process	NP-Complete (performance is heuristic-dependent)	Suffers from path explosion; heuristics can be brittle and fail on adversarial instances. [69]
Generative Inversion (OGY-RHA)	Dynamical Control on Attractor Landscape	Final State + Spectral Residue Analysis	Potentially sub-exponential for systems with detectable residues	Requires the existence of detectable residues and a sufficient understanding of the system's attractor dynamics.

Discussion and Future Directions

The Recursive Harmonic Architecture offers a new lens through which to view computation, with profound implications for cryptography, artificial intelligence, and our understanding of physical law.

Implications for a New Kind of Cryptography and AI

The RHA framework suggests a path toward "tunably secure" cryptographic functions. By adjusting the system parameters that govern the chaotic dynamics—such as the coupling strength in the genesis lattice or the specific choice of non-linear functions—one could precisely control the prominence and complexity of the spectral residues. This would allow for the design of cryptographic primitives where inversion difficulty is not merely an unproven assumption but a predictable, designable parameter, directly tied to the system's susceptibility to chaos control techniques.[13]

For artificial intelligence and A-Life, the genesis lattice represents a significant step toward truly autonomous, generative agents. Such a system is not just executing a program; it is actively maintaining its own existence through autopoiesis.[52] Its ability to invert its own generative process via chaos control constitutes a form of computational self-reflection. This could provide a foundation for AI systems capable of explaining their own reasoning not through post-hoc rationalization, but by dynamically reversing their generative "thought" process to reveal the trajectory that led to a conclusion.

Connections to Fundamental Physics and Information Theory

The principles underlying the RHA resonate with deep concepts in theoretical physics. A speculative but compelling connection exists with the **holographic principle**, which posits that the information describing a volume of space is encoded on its lower-dimensional boundary.[76] In the RHA, the final state (the hash) and its associated spectral residues can be viewed as a low-dimensional projection that encodes the full informational content of the higher-dimensional computational history—the complete space-time evolution of the genesis lattice. Generative inversion, in this light, is analogous to reconstructing the "bulk" physics from the "boundary" information.

Furthermore, the rule set of the genesis lattice need not be static. Drawing inspiration from theorists like Lee Smolin, who propose that the laws of physics may themselves evolve over time, one can envision a meta-level dynamic. In such a system, the parameters and rules of the RHA could be subjected to evolutionary pressures, leading to a computational universe with its own evolving "physics."

Future Work and Open Questions

This paper lays a theoretical foundation, but significant experimental and theoretical work remains.

- **Large-Scale Simulation:** A primary avenue for future work is the simulation of the genesis lattice on next-generation, AI-focused supercomputing architectures. Systems like the **Nexus** supercomputer at Georgia Tech, designed for massive parallelism and handling giant datasets, are ideal platforms for exploring the emergent dynamics and inversion properties of large-scale RHA implementations.[78]
- **Evolutionary Design of Rule Sets:** The vast space of possible RHA rule sets can be explored using evolutionary algorithms.[80] By defining fitness functions based on desired properties—such as specific spectral residue signatures, levels of computational irreducibility, or inversion complexity—one could evolve novel computational "physics" tailored for specific tasks.
- **Formalizing the Residue-Control Link:** A more rigorous mathematical theory is needed to formally connect the statistical properties of spectral residues to the control parameters of the OGY-RHA algorithm. This would involve moving from the strong analogy with OGY to a formal proof of controllability for this class of discrete, high-dimensional chaotic systems, likely requiring tools from proof theory and reverse mathematics.[81]

Works cited

1. Models of Computation, accessed July 27, 2025, https://cs.brown.edu/people/jsavage/book/pdfs/ModelsOfComputation.pdf
2. Scientists Think the Universe Is a Quantum Computer - Here's The Physics Behind It!, accessed July 26, 2025, https://www.youtube.com/watch?v=PONu9_8rFfI
3. The Computational Universe | American Scientist, accessed July 26, 2025, https://www.americanscientist.org/article/the-computational-universe
4. Artificial life - Wikipedia, accessed July 27, 2025, https://en.wikipedia.org/wiki/Artificial_life
5. Artificial Life as Theoretical Biology: How to do real science with computer simulation - Geoffrey Miller, accessed July 27, 2025, https://geoffrey-miller-y5jr.squarespace.com/s/1995-real-science-simulation.pdf
6. Computing the Origin of Life | News - NASA Astrobiology, accessed July 27, 2025, https://astrobiology.nasa.gov/news/computing-the-origin-of-life/
7. (PDF) Thirty Years of Computational Autopoiesis: A Review, accessed July 27, 2025, https://www.researchgate.net/publication/8462896_Thirty_Years_of_Computational_Autopoiesis_A_Review
8. Rediscovering Computational Autopoiesis - CiteSeerX, accessed July 27, 2025, https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=02080ab1602242fc9e9bbf4560b65bb9192b29f7
9. (PDF) Rediscovering Computational Autopoiesis - ResearchGate, accessed July 27, 2025, https://www.researchgate.net/publication/2831744_Rediscovering_Computational_Autopoiesis
10. (PDF) Cryptographic Hash Functions: Recent Design Trends and Security Notions., accessed July 27, 2025, https://www.researchgate.net/publication/220334031_Cryptographic_Hash_Functions_Recent_Design_Trends_and_Security_Notions
11. Hash Function - Definitions, Example, How it Works - Corporate Finance Institute, accessed July 27, 2025, https://corporatefinanceinstitute.com/resources/cryptocurrency/hash-function/
12. Cryptographic Hash Functions – Networks at ITP, accessed July 27, 2025, https://itp.nyu.edu/networks/explanations/cryptographic-hash-functions/
13. TechnicalExperts/writing/computational_irreducibility.md at main - GitHub, accessed July 27, 2025, https://github.com/Jason2Brownlee/TechnicalExperts/blob/main/writing/computational_irreducibility.md
14. Computational irreducibility - Wikipedia, accessed July 27, 2025, https://en.wikipedia.org/wiki/Computational_irreducibility
15. Hash functions using chaotic iterations - arXiv, accessed July 27, 2025, https://arxiv.org/pdf/1702.02489
16. New hash function based on chaos theory (CHA-1) - ResearchGate, accessed July 27, 2025, https://www.researchgate.net/publication/253155858_New_hash_function_based_on_chaos_theory_CHA-1

17. Are hash functions chaotic? - Cryptography Stack Exchange, accessed July 27, 2025, https://crypto.stackexchange.com/questions/16144/are-hash-functions-chaotic
18. Is computational irreducibility a real thing? : r/math - Reddit, accessed July 27, 2025, https://www.reddit.com/r/math/comments/1c53wna/is_computational_irreducibility_a_real_thing/
19. Computational requirements for breaking SHA-256? - Cryptography Stack Exchange, accessed July 27, 2025, https://crypto.stackexchange.com/questions/52571/computational-requirements-for-breaking-sha-256
20. SHA-256 Algorithm: Characteristics, Steps, and Applications - Simplilearn.com, accessed July 27, 2025, https://www.simplilearn.com/tutorials/cyber-security-tutorial/sha-256-algorithm
21. Merkle–Damgård construction - Wikipedia, accessed July 27, 2025, https://en.wikipedia.org/wiki/Merkle%E2%80%93Damg%C3%A5rd_construction
22. Merkle-Damgård as the Foundation of Hash Cryptography: A Study of Advantages and Limitations, accessed July 27, 2025, https://jurnal.seaninstitute.or.id/index.php/juti/article/download/652/417
23. SHA-256 Collision Attack with Programmatic SAT - arXiv, accessed July 27, 2025, https://arxiv.org/html/2406.20072v1
24. What is the SHA-256 Cryptographic Hash Algorithm? - SSLInsights, accessed July 27, 2025, https://sslinsights.com/sha-256-cryptographic-hash-algorithm/
25. How Cryptographic Hash Functions Solve a Very Difficult and Important Problem, accessed July 27, 2025, https://bennettgarner.medium.com/how-cryptographic-hash-functions-solve-a-very-difficult-and-important-problem-b939da3b0185
26. Cryptographic Hash Functions: A Historical Overview - Freeman Law, accessed July 27, 2025, https://freemanlaw.com/cryptographic-hash-functions/
27. A Deep Dive into SHA-256: Working Principles and Applications | by Madan | Medium, accessed July 27, 2025, https://medium.com/@madan_nv/a-deep-dive-into-sha-256-working-principles-and-applications-a38cccc390d4
28. SHA-256 Under the Hood. Look inside the popular hash function. | Medium, accessed July 27, 2025, https://medium.com/@PicKeyAI/sha-256-under-the-hood-83e332c468ef
29. fips pub 180-4 - federal information processing standards publication, accessed July 27, 2025, https://nvlpubs.nist.gov/nistpubs/fips/nist.fips.180-4.pdf
30. Attractor - Wikipedia, accessed July 27, 2025, https://en.wikipedia.org/wiki/Attractor
31. Chaos theory - Wikipedia, accessed July 27, 2025, https://en.wikipedia.org/wiki/Chaos_theory
32. Attractor Basins, Tipping Points · DynamicalSystems.jl - GitHub Pages, accessed July 27, 2025, https://juliadynamics.github.io/DynamicalSystems.jl/previews/PR156/chaos/basins/
33. Discrete Dynamical Networks and their Attractor Basins, accessed July 27, 2025, http://ddlab.org/downloads/papers/complex98.pdf
34. Demonstrate the Avalanche Effect of a Hash Function: New in ..., accessed July 27, 2025, https://www.wolfram.com/language/12/cryptography/demonstrate-the-avalanche-effect-of-a-hash-function.html

35. Byte1 and the π Lattice: A Unified Interface-Driven Recursion, accessed July 27, 2025, https://zenodo.org/records/15825312
36. (PDF) Practical Electromagnetic Template Attack on HMAC - ResearchGate, accessed July 27, 2025, https://www.researchgate.net/publication/221291921_Practical_Electromagnetic_Template_Attack_on_HMAC
37. aes - Hamming Distance and Avalanche effect in Cryptography ..., accessed July 27, 2025, https://crypto.stackexchange.com/questions/108423/hamming-distance-and-avalanche-effect-in-cryptography
38. Differential Cryptanalysis Tutorial - The Amazing King, accessed July 27, 2025, http://www.theamazingking.com/crypto-diff.php
39. (PDF) Improving Local Collisions: New Attacks on Reduced SHA-256, accessed July 27, 2025, https://www.researchgate.net/publication/235257652_Improving_Local_Collisions_New_Attacks_on_Reduced_SHA-256
40. 22 step linear characteristic for SHA-256. There are 59 GH local collisions. - ResearchGate, accessed July 27, 2025, https://www.researchgate.net/figure/22-step-linear-characteristic-for-SHA-256-There-are-59-GH-local-collisions_tbl5_220335111
41. Error Correction Capabilities of Non-Linear Cryptographic Hash Functions - arXiv, accessed July 27, 2025, https://arxiv.org/html/2405.01495v1
42. Analysis of simplified variants of SHA-256 - EMIS, accessed July 27, 2025, https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=3ff2da899ef30dba6000ef86d1794127b2d49f20
43. On Various Nonlinearity Measures for Boolean Functions - PMC, accessed July 27, 2025, https://pmc.ncbi.nlm.nih.gov/articles/PMC4957665/
44. Analysis of Building Blocks in SHA-256 - Maxwell Science, accessed July 27, 2025, https://maxwellsci.com/print/rjaset/10-15.pdf
45. Show HN: SHA-256 explained step-by-step visually | Hacker News, accessed July 27, 2025, https://news.ycombinator.com/item?id=30244534
46. Nothing-up-my-sleeve number - Wikipedia, accessed July 27, 2025, https://en.wikipedia.org/wiki/Nothing-up-my-sleeve_number
47. Nothing up my sleeve numbers - cryptologie.net, accessed July 27, 2025, https://www.cryptologie.net/article/221/nothing-up-my-sleeve-numbers/
48. Nothing-up-my-sleeve numbers standard - Cryptography Stack Exchange, accessed July 27, 2025, https://crypto.stackexchange.com/questions/87638/nothing-up-my-sleeve-numbers-standard
49. The cryptographic hash function SHA-256, accessed July 27, 2025, https://helix.stormhub.org/papers/SHA-256.pdf
50. How does the SHA256 algorithm work…in detail? (part 1/2) | by Nicky Reinert - Medium, accessed July 27, 2025, https://nickyreinert.medium.com/how-does-the-sha256-algorithm-in-detail-part-1-2-45154fab02d2

51. What does "message schedule" mean in SHA-256? - Cryptography Stack Exchange, accessed July 27, 2025, https://crypto.stackexchange.com/questions/8636/what-does-message-schedule-mean-in-sha-256

52. (PDF) Computing with Autopoietic Systems - ResearchGate, accessed July 27, 2025, https://www.researchgate.net/publication/254842986_Computing_with_Autopoietic_Systems

53. A Study of "Organizational Closure" and Autopoiesis: | Harish's Notebook, accessed July 27, 2025, https://harishsnotebook.wordpress.com/2019/07/21/a-study-of-organizational-closure-and-autopoiesis/

54. Maturana's Autopoiesis in AI: Self-Creation Through Recursive Organization - Reddit, accessed July 27, 2025, https://www.reddit.com/r/ArtificialSentience/comments/1l5qhcs/maturanas_autopoiesis_in_ai_selfcreation_through/

55. A Concise and Formal Definition of RAF Sets and the RAF Algorithm, accessed July 27, 2025, https://arxiv.org/pdf/2303.01809

56. Conway's Game of Life - Wikipedia, accessed July 27, 2025, https://en.wikipedia.org/wiki/Conway%27s_Game_of_Life

57. Coupled Map Lattice - Nonlinear Systems - WordPress.com, accessed July 27, 2025, https://nonlinearsystems.wordpress.com/2021/03/24/coupled-map-lattice/

58. Mastering SHA-256: The Ultimate Guide - Number Analytics, accessed July 27, 2025, https://www.numberanalytics.com/blog/mastering-sha-256-ultimate-guide

59. SHA-256 Collision Attack with Programmatic SAT - CEUR-WS.org, accessed July 27, 2025, https://ceur-ws.org/Vol-3717/paper5.pdf

60. Extending SAT Solvers to Cryptographic Problems, accessed July 27, 2025, https://www.msoos.org/wordpress/wp-content/uploads/2011/03/Extending_SAT_2009.pdf

61. CDCL(Crypto) SAT Solvers for Cryptanalysis - arXiv, accessed July 27, 2025, https://arxiv.org/pdf/2005.13415

62. Control of chaos - Wikipedia, accessed July 27, 2025, https://en.wikipedia.org/wiki/Control_of_chaos

63. Control of Chaotic Dynamical Systems using OGY - IS MUNI, accessed July 27, 2025, https://is.muni.cz/el/1431/jaro2016/M6201/um/OGY.pdf

64. Controlling Chaotic Dynamical Systems - SciSpace, accessed July 27, 2025, https://scispace.com/pdf/controlling-chaotic-dynamical-systems-w26kfvwcmz.pdf

65. Learning Rate Based Branching Heuristic for SAT Solvers | Request PDF - ResearchGate, accessed July 27, 2025, https://www.researchgate.net/publication/303901008_Learning_Rate_Based_Branching_Heuristic_for_SAT_Solvers

66. Boosting the Performance of CDCL-Based SAT Solvers by Exploiting Backbones and Backdoors - MDPI, accessed July 27, 2025, https://www.mdpi.com/1999-4893/15/9/302

67. Modern SAT solvers: fast, neat and underused (part 3 of N) - The Coding Nest, accessed July 27, 2025, https://codingnest.com/modern-sat-solvers-fast-neat-and-underused-part-3-of-n/

68. Theoretical explanations for practical success of SAT solvers?, accessed July 27, 2025, https://cstheory.stackexchange.com/questions/37886/theoretical-explanations-for-practical-success-of-sat-solvers

69. Learning to Explore Paths for Symbolic Execution, accessed July 27, 2025, https://files.sri.inf.ethz.ch/website/papers/ccs21-learch.pdf

70. NEURAL HEURISTICS FOR SAT SOLVING - Representation Learning on Graphs and Manifolds, accessed July 27, 2025, https://rlgm.github.io/papers/32.pdf

71. Crypto-SAT - Google Sites, accessed July 27, 2025, https://sites.google.com/view/crypto-sat/

72. [2406.20072] SHA-256 Collision Attack with Programmatic SAT - arXiv, accessed July 27, 2025, https://arxiv.org/abs/2406.20072

73. SHA-256 Collision Attack with Programmatic SAT, accessed July 27, 2025, https://cs.uwaterloo.ca/~cbright/reports/sc2-hash.pdf

74. Scaling Symbolic Execution to Large Software Systems - arXiv, accessed July 27, 2025, https://arxiv.org/html/2408.01909v1

75. Artificial Intelligence is Algorithmic Mimicry: Why artificial “agents” are not (and won't be) proper agents - arXiv, accessed July 27, 2025, https://arxiv.org/html/2307.07515v4

76. Holographic Principle in Quantum Computing - Number Analytics, accessed July 27, 2025, https://www.numberanalytics.com/blog/holographic-principle-quantum-computing-ads-cft

77. Holographic principle - Wikipedia, accessed July 27, 2025, https://en.wikipedia.org/wiki/Holographic_principle

78. Georgia Tech to Build $20M National AI Supercomputer | News Center, accessed July 26, 2025, https://news.gatech.edu/news/2025/07/15/georgia-tech-build-20m-national-ai-supercomputer

79. NEXUS: a computer language for physiological systems and signal analysis - PubMed, accessed July 26, 2025, https://pubmed.ncbi.nlm.nih.gov/6548943/

80. Genetic algorithm - Wikipedia, accessed July 27, 2025, https://en.wikipedia.org/wiki/Genetic_algorithm

81. Reverse mathematics - Wikipedia, accessed July 27, 2025, https://en.wikipedia.org/wiki/Reverse_mathematics

82. Formalism 25 - arXiv, accessed July 27, 2025, https://arxiv.org/html/2502.14811v2