Biological Motifs for Agentic Control

A Categorical Isomorphism between Gene Regulatory Networks and Autonomous Software Architectures

Preprint – Feedback Welcome

Bogdan Banu

`bogdan@banu.be`

December 26, 2025

Abstract

The transition of Large Language Models (LLMs) from passive generators to autonomous agents has introduced significant challenges in reliability, security, and state management. Current agentic architectures are often constructed ad-hoc, prone to "hallucination cascades," infinite loops, and prompt injection attacks. This paper proposes that these failure modes are not unique to software but are instances of universal control problems solved by biological systems over billions of years.

We present a formal isomorphism, *at the level of their polynomial-interface models*, between Gene Regulatory Networks (GRNs) and Agentic Software Systems using Applied Category Theory. We model agents as Polynomial Functors within the category **Poly**, and their interactions via the Operad of Wiring Diagrams. We derive a rigorous syntax for agent composition by mapping biological mechanisms—including Quorum Sensing for consensus, Chaperone Proteins for structural validation, and Endosymbiosis for neuro-symbolic integration—to software design patterns. This framework provides a mathematical basis for "Epigenetic" state management (RAG) and the topological defense against adversarial "Prion" attacks.

1 Introduction

The field of Artificial Intelligence is undergoing a paradigm shift from Generative AI (systems that produce text based on static prompts) to Agentic AI (systems that execute multi-step workflows to achieve autonomous goals). While the capabilities of individual Large Language Models (LLMs) have scaled predictably, the engineering of systems of agents remains a fragile art. Developers struggle with non-deterministic outputs, infinite loops, adversarial attacks, and the difficulty of maintaining global coherence in distributed, stochastic systems.

We argue that these challenges are not novel engineering problems, but fundamental constraints of distributed information processing systems. The closest existing analogue to a multi-agent software architecture is not a traditional computer program, but a Gene Regulatory Network (GRN). In a biological cell, thousands of genes act as autonomous agents, reading local chemical signals (context) and expressing proteins (actions/tools) that, in turn, regulate other genes.

1.1 The Biological Heuristic

Biology has evolved specific topological structures, known as Network Motifs, to handle noise, security, and state [6]. We identify four critical biological heuristics that map directly to agentic engineering:

- The Coherent Feed-Forward Loop (CFFL): Acts as a persistence detector to filter out transient noise, analogous to "Human-in-the-Loop" guardrails.

- Quorum Sensing: A distributed consensus mechanism where action is taken only when signal density exceeds a threshold, analogous to Mixture of Experts (MoE) voting.

- Chaperone Proteins: Molecular cages that force proteins to fold correctly, analogous to Schema Validators that enforce structured outputs (JSON).

- Immunological Self-Defense: Mechanisms to distinguish self from non-self, relevant to preventing Prompt Injection attacks.

1.2 The Categorical Bridge

To move this observation from metaphor to discipline, we utilize Applied Category Theory. We define the category of agents using the language of **Poly** (Polynomial Functors) as described by Spivak [8]. An agent is not defined by its weights, but by its interface—a dynamical system consuming observations and producing actions:

$$P_A(y) = \sum_{o \in O} y^{I_o}. \tag{1}$$

1.3 Contributions

This paper makes the following contributions:

1. **A Formal Dictionary:** We establish a rigorous mapping between biological components (Genes, Promoters, Plasmids) and software components (Agents, Schemas, Tools).

2. **The Agentic Operad:** We define WAgent, a syntax for agent wiring that forbids specific classes of **ill-typed wirings** (and thus their associated runtime **type/schema mismatch** errors) at the topological level.

3. **Pathology Identification:** We classify agentic failures as biological diseases, mapping Infinite Loops to Cancer, Hallucinations to Autoimmunity, and Prompt Injections to Prion Disease.

4. **Future Architectures:** We propose Endosymbiosis as a model for Neuro-Symbolic AI, where LLMs "engulf" deterministic runtimes to gain computational energy.

   By viewing agentic engineering through the lens of theoretical biology and category theory, we aim to provide a foundation for building robust software systems whose stability properties derive from their network topology.

2 Related Work

This work sits at the intersection of Systems Biology, Applied Category Theory, and Agentic AI. While significant research exists within each domain, the formal synthesis of biological control topologies with agentic software architectures has received limited attention.

2.1 Network Motifs in Systems Biology

The concept of "Network Motifs"—statistically over-represented sub-graphs in complex networks—was introduced by Milo et al. [6]. Their work demonstrated that biological networks are not random but are composed of specific building blocks selected for functional data processing. Alon [1] further characterized the dynamical properties of these motifs, identifying the Coherent Feed-Forward Loop (CFFL) as a persistence detector. We extend this by mapping these motifs to the stochastic nature of Generative AI.

2.2 Applied Category Theory (ACT)

To formalize network structure, we draw upon ACT. Spivak [8] and Vagner et al. [9] established a rigorous framework for modeling Open Dynamical Systems using the category **Poly** and the Operad of Wiring Diagrams. To our knowledge, this is the first application of Polynomial Functors specifically designed to model the interface of LLM Agents and to verify safety properties in Agentic topologies.

2.3 Reliability in Agentic AI

Techniques such as "Chain of Thought" [10] utilize iterative looping to improve output quality. However, these methods operate primarily at the level of the prompt (the input signal) rather than the topology (the wiring). By importing the concept of Autopoiesis [5], we propose a methodology where reliability is a property of the network architecture itself.

3 The Mapping: Biology ↔ Software

To treat Agentic Systems and Gene Regulatory Networks (GRNs) as isomorphic **at the level of their typed interfaces**, we must map them to a common mathematical object. We utilize the category **Poly**, where objects are polynomial functors representing interfaces, and morphisms represent interaction protocols.

3.1 Preliminaries: The Category Poly

In Applied Category Theory, a Polynomial Functor P represents a typed interface for a dynamical system. It is defined as a sum of representable functors:

$$P(y) = \sum_{o \in O} y^{I_o}. \tag{2}$$

Here, O is the set of possible Positions (or Outputs) the system can expose. For each position $o \in O$, there is a set I_o of Directions (or Inputs) required to transition the system to a new state.

- The coefficient o represents the value produced by the system.

- The exponent I_o represents the capacity to receive information from the environment.

This formalism captures the essence of a "stateful interface": the system outputs a value o and then waits for a specific type of input $i \in I_o$ before it can proceed.

3.2 The Isomorphism: Genes and Agents

We now apply this abstract definition to our specific domains.
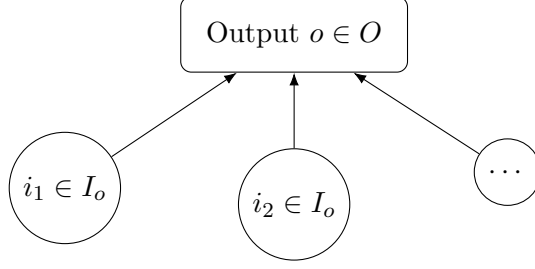
The Interface $P(y)$

Figure 1: A visual representation of a Polynomial Functor (often called a "Mushroom" or "Corolla"). The system offers an Output (the cap) and exposes specific Input ports (the stalks) dependent on that output.

Definition 1 (The Gene Object). A gene G is a polynomial functor where O_G is the set of expressed proteins and $I_G = (I_{\mathrm{prot}})_{\mathrm{prot} \in O_G}$ is the **family** of regulatory-signal sets (transcription factors) available at each expressed protein:

$$P_{\mathrm{Gene}}(y) = \sum_{\mathrm{prot} \in \mathrm{Proteins}} y^{I_{\mathrm{prot}}}. \tag{3}$$

Definition 2 (The Agent Object). An autonomous agent A is a polynomial functor where O_A is the set of generated messages/actions, and $I_A = (I_{\mathrm{action}})_{\mathrm{action} \in O_A}$ is the **family** of observation sets available at each action:

$$P_{\mathrm{Agent}}(y) = \sum_{\mathrm{action} \in \mathrm{Actions}} y^{I_{\mathrm{action}}}. \tag{4}$$

3.3 The Interface: Promoters as Lenses

In biology, a gene is not universally accessible. It is guarded by a Promoter Region—a specific DNA sequence that only binds to compatible Transcription Factors. In software, an agent is guarded by an API Schema or Context Window definition.

We model this gating mechanism using Optics, specifically Lenses. A Lens consists of two maps between a global state S and a local view V:

1. Get (View): $\mathrm{get} : S \to V$ (Extracting relevant signal from global state).

2. Put (Update): $\mathrm{put} : S \times V' \to S$ (Updating global state based on local change).

The "Promoter" acts as a filter that determines which part of the global cellular environment (S) is visible (V) to the gene.

- **Biological Lens:** The promoter filters the chaotic cellular soup, allowing the gene to "see" only specific molecules (e.g., Lac Repressor).

- **Agentic Lens:** The Context Window filters the massive vector database, allowing the agent to "see" only the relevant retrieved chunks (RAG).

4

If the input signal does not match the Schema (Promoter), the Lens fails to focus, and the interaction is routed to an explicit **inactive/error** case (equivalently, one works with a **partial lens**, or a total lens into $V + \text{Error}$) (the agent does not run; the gene is not expressed).

3.4 Epigenetics and State: The Coalgebra

Neither genes nor agents are stateless functions. They possess memory.

- **Biology:** Epigenetic markers (Methylation, Histone modification) alter how a gene responds to signals without changing the DNA code.

- **Software:** Retrieval Augmented Generation (RAG) and Conversation History alter how an agent responds to a prompt without changing the LLM weights.

We model this as a Coalgebra for the polynomial functor P. A dynamical system is defined as a tuple (S, ϕ), where S is the state space and ϕ is the structure map:

$$\phi : S \to P(S). \tag{5}$$

By expanding $P(S)$, we derive the two fundamental operations of the state machine:

1. Readout: $S \to O$ (Given current state/memory, what action do I take?)

2. Update: $\sum_{s \in S} I_{o(s)} \to S$ (Given current state s and a new input $i \in I_{o(s)}$ compatible with its current output $o(s)$, what is my new state?)

By establishing this formal dictionary (Table 1), we **can regard** GRNs and Agentic Systems as distinct implementations of the same abstract dynamical system **under this interface-level abstraction**.

Gene (Function)	Agent (LLM + Tools)
Transcription Factors (I)	Observations (I)
Proteins (O)	Actions (O)
Promoter Binding	Schema Match
Expression	Generation

Figure 2: The Structural Isomorphism. Both Genes and Agents act as transducers converting Input Contexts (I) into Output Expressions (O), governed by the same categorical laws (at the level of polynomial-interface models).

3.5 Metabolic Coalgebras: Formalizing Resource Constraints

Finally, we address the physical constraints of computation. Just as biological systems are limited by ATP availability [4], agentic systems are limited by token budgets and latency. To model this, we extend our coalgebraic framework to include resource constraints, defining a **Metabolic Coalgebra**. Mathematically, this is an instance of a Quantitative Coalgebra enriched over a resource monoid, effectively restricting the domain of the state transition function to resource-sufficient states.

We align this definition with the theory of **Quantitative Polynomial Functors** [7], treating the system as a state machine enriched over a resource monoid.

Category Concept	Biological Realization (GRN)	Software Realization (Agentic)
Polynomial Functor (P)	Gene Interface	Agent Interface (System Prompt)
Output Position (O)	Protein Expression	Tool Call / Message
Input Direction (I)	Transcription Factor Binding	Observation / User Prompt
Lens (Optic)	Promoter Region	API Schema / Context Window
Internal State (S)	Epigenetic Markers (Methylation)	Vector Store / Chat History
Morphism ($\circ$)	Signal Transduction Pathway	Data Pipeline

Table 1: The Isomorphism Dictionary

Definition 1 (The Resource Monoid). *Let $(\mathcal{R}, +, 0, \geq)$ be an ordered commutative monoid representing computational resources (e.g., token counts), where $\mathcal{R} \cong \mathbb{N}$.*

Definition 2 (Metabolic Coalgebra). *A resource-constrained agent is a coalgebra (S, α) over a polynomial functor P, where the state space is the product of the logical state L and the resource state $\mathcal{R}$:*

$$S \cong L \times \mathcal{R} \tag{1}$$

*The structure map $\alpha : S \to P(S) + \perp$ is defined as a **partial map** guarded by cost. For a transition requiring cost $c \in \mathcal{R}$:*

$$\alpha(l, r) = \begin{cases} (l', r - c) & \text{if } r \geq c \\ \perp & \text{if } r < c \quad \text{(Apoptosis)} \end{cases} \tag{2}$$

This structure maps to the energetics of transcriptional elongation. A gene (Agent) cannot express its protein (Action) instantaneously; it must transcribe an mRNA sequence (Chain of Thought) nucleotide by nucleotide. This process consumes a distinct amount of chemical energy (NTPs) per step. The Metabolic Coalgebra models this dependency: if the cellular energy budget is exhausted, transcription stalls (Ischemia), and the gene fails to execute its function, regardless of its regulatory logic.

This formalism establishes that "Ischemia" (Token Starvation) is not merely a runtime error, but a reachable terminal state $\perp$ in the system's dynamics. This mirrors the biological mechanism where failure to meet metabolic costs triggers p53-mediated apoptosis [2].

Theorem 1 (The Metabolic Bound). *For any agentic topology T composed of N agents with total budget R_{total}, the system is guaranteed to halt. Unlike the general Halting Problem, termination is decidable for Metabolic Coalgebras: the resource state r is strictly decreasing for every non-identity morphism, providing a well-founded termination measure [3].*

4 Formal Syntax: The Agentic Operad

To formalize the composition of agents, we define the Operad of Wiring Diagrams, denoted as WAgent. An operad can be understood as a "grammar" for connecting operations (boxes) via typed wires. It defines which agent topologies are valid and allows us to reason about the properties of the composite system based solely on the properties of its components [9].

4.1 The Typing Rules

In WAgent, every wire carries a specific Type $\tau \in T$.

$$T = \{\text{Text}, \text{JSON}, \text{Image}, \text{Error}, \text{ToolCall}\}. \tag{6}$$

These types correspond to biological molecular specificity (e.g., a specific transcription factor only binds to a specific DNA sequence). A connection is valid if and only if the type of the output port of Agent A matches the type of the input port of Agent B.

4.2 The Composition Operations

The operad defines three fundamental operations for combining agents. Any complex agentic architecture, no matter how large, can be decomposed into these three primitives.

4.2.1 Parallel Composition ($\otimes$)

Two agents, A and B, execute simultaneously with no information exchange:

$$A \otimes B. \tag{7}$$

- **Biological Analogy:** Two genes located on different chromosomes expressing proteins independently.

- **Constraint:** This operation is valid only if the internal state spaces S_A and S_B are disjoint. If they share a mutable memory store, the operation leaves the **independent-state interpretation** and requires an explicit **resource-sharing** structure (e.g., a shared state component or a Resource Sharing decorator).

4.2.2 Serial Composition ($\circ$)

The output of Agent A is piped directly into the input of Agent B:

$$B \circ A. \tag{8}$$

- **Biological Analogy:** A Signal Transduction Pathway (Protein A activates Protein B).

- **Formal Verification:** This allows for static type checking of agent graphs. If Agent A outputs Natural Language but Agent B expects JSON Schema, the composition is undefined in WAgent. This moves runtime **type/schema mismatch** errors to "compile-time" architectural errors.

4.2.3 Contraction / Trace (Tr)

A feedback loop where an output port of Agent A is wired back into one of its own input ports:

$$Tr(A). \tag{9}$$

- **Biological Analogy:** Autoregulation (Homeostasis) or Positive Feedback.

- **Software Implication:** This is the topological definition of Agency. A "stateless" LLM is a simple morphism. An "Agent" is a morphism wrapped in a Trace operation, allowing it to observe its own previous outputs (Chain-of-Thought).

4.3 Theorem: Topological Error Suppression

We now use this formalism to show that the Coherent Feed-Forward Loop (CFFL) provides stronger error suppression guarantees than a direct connection for high-stakes tasks.

Network Motif 1 (Coherent Feed-Forward Loop). A topological structure where Signal X activates Z directly, but also activates Y which gates Z. The node Z functions as an AND gate: it fires if and only if $X \wedge Y$.

Theorem 1 (Error Suppression in CFFL). Let A_{gen} be a generator agent and A_{ver} be a verifier agent. Let $P(E_{\text{gen}})$ (resp. $P(E_{\text{ver}})$) be the probability of a hallucination (error) in any single generation step of A_{gen} (resp. A_{ver}).

- **Case 1: Direct Link (Serial).** The system fails if A_{gen} hallucinates.

$$P(\text{Fail}_{\text{direct}}) = P(E_{\text{gen}}).$$

- **Case 2: CFFL Topology.** The system requires the logical conjunction of the Generator's request (X) and the Verifier's approval (Y). Assuming the Verifier's error mode is independent of the Generator's (e.g., different prompt strategy or model temperature), the probability that both agents simultaneously hallucinate a "Go" signal for a destructive action is:

$$P(\text{Fail}_{\text{CFFL}}) = P(E_{\text{gen}} \wedge E_{\text{ver}}) = P(E_{\text{gen}}) \times P(E_{\text{ver}}).$$

Since $0 \leq P(E_{\text{gen}}), P(E_{\text{ver}}) \leq 1$, it follows that

$$P(E_{\text{gen}})P(E_{\text{ver}}) \leq \min\{P(E_{\text{gen}}), P(E_{\text{ver}})\},$$

with strict inequality whenever both probabilities lie in $(0, 1)$.

Proof. In WAgent, the CFFL is defined as a morphism involving a "Copy" operation $\Delta_X : X \to X \otimes X$ and an "AND-Merge" operation $\mu : Z \otimes Y \to \text{Out}$. The existence of the μ box in the wiring diagram structurally enforces **conjunctive gating**; statistical independence (or low correlation) between error modes is an additional modeling assumption that can be encouraged by diversity in prompts/models. This shows that certain safety properties are a property of the topology, not just the prompt engineering.
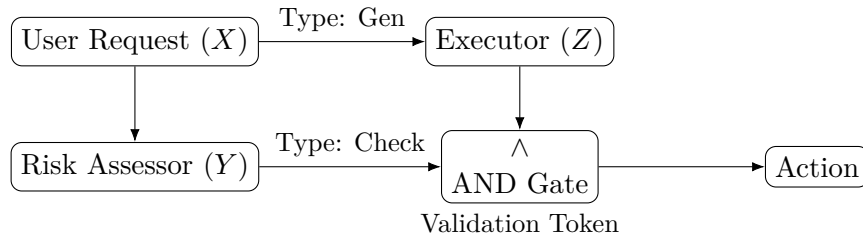

Figure 3: The CFFL implemented in WAgent. The Executor (Z) cannot act without the token from the Risk Assessor (Y), topologically preventing unilateral execution without approval.

4.4 Quorum Sensing (Consensus & Voting)

Network Motif 2 (Quorum Sensing). A distributed topology where multiple agents emit a weak signal σ into a shared environment. An effector node E activates if and only if the concentration $[\sigma] > \theta$.

- **Biological Function:** Many bacteria (e.g., *V. fischeri*) secrete auto-inducer molecules. Individual bacteria do not react to low concentrations. However, once the population density reaches a threshold (Quorum), the concentration of auto-inducers triggers a simultaneous, coordinated gene expression event (e.g., bioluminescence or biofilm formation).

- **Agentic Isomorphism (Voting Ensembles):** In non-deterministic systems, a single agent's output is noisy. By instantiating N parallel agents (a Mixture of Experts), the system aggregates their outputs. The final action is taken only if the "concentration" of a specific semantic token exceeds a confidence threshold. This transforms weak, noisy individual signals into a robust, high-confidence collective action.

4.5 Chaperone Proteins: Output Structural Validation

- **Biological Function:** Newly synthesized proteins emerge as linear chains that must fold into precise 3D structures to function. Chaperone Proteins (e.g., GroEL-GroES) sequester unfolded proteins, preventing aggregation and facilitating correct folding. If a protein fails to fold repeatedly, it is tagged for degradation (Ubiquitination) to prevent toxic buildup.

- **Agentic Isomorphism (Retry & Repair Loops):** Generative models output unstructured token streams ("linear chains"). However, downstream agents require strictly structured inputs (e.g., valid JSON Schemas). A Validator Agent acts as a Chaperone: it intercepts the raw output, attempts to parse it into a formal schema ("folding"), and if validation fails, returns the error trace to the generator for re-synthesis. This turns a probabilistic string into a deterministic data structure.

- **Categorical View:** The Chaperone acts as a **partial** retraction: there is an inclusion $i : V \to S$ and a map $r : S \to V + \text{Error}$ such that $r \circ i = \text{inl} \circ \text{id}_V$, and r returns Error on ill-formed text.

5 Failure Modes & Pathology

A key insight of Systems Biology is that diseases are often not caused by the complete failure of a single component, but by the dysregulation of network dynamics. A cancerous cell still "works"—in fact, it works too well, reproducing indefinitely. Similarly, catastrophic failures in agentic systems often arise from functional agents interacting in topologically pathological ways.

We classify three primary classes of agentic pathology based on their biological isomorphisms.

5.1 Oncology: Infinite Loops as Unchecked Growth

- **Biological Pathology (Cancer):** In a healthy cell, the cell cycle is driven by a positive feedback loop (Cyclins) but constrained by negative feedback "checkpoints" (e.g., the p53 gene). If p53 is mutated, the negative feedback is severed. The positive loop runs unchecked, leading to exponential proliferation (tumor growth).

- **Agentic Pathology (The Recursive Hang):** Two agents get stuck in a politeness loop (e.g., "Thank you," "You're welcome") or a debugger agent continuously generates new bugs to fix old ones.

- **Categorical Diagnosis:** The Trace operation $Tr(A)$ is not equipped with a **well-founded termination measure**; equivalently, the induced state transition may admit cycles, and under

an information-gain metric d (e.g., $1-$similarity of successive messages) the distance between state S_t and S_{t+1} can approach zero while the system fails to trigger a "Stop" token.

- **Treatment:** Implementation of an Apoptosis (Cell Death) mechanism. A meta-monitor agent observes the entropy of the conversation. If information gain drops below a threshold (the conversation becomes repetitive), the monitor forces a termination signal.

5.2 Autoimmunity: Hallucination Cascades

- **Biological Pathology (Autoimmune Disease):** The immune system relies on distinguishing "Self" (internal tissue) from "Non-Self" (foreign pathogens). In diseases like Lupus, this distinction blurs, and the system attacks healthy tissue.

- **Agentic Pathology (Context Poisoning):** Agent A hallucinates a fact (e.g., a non-existent library function). Agent B reads this hallucination from the shared history, treats it as ground truth, and builds complex logic upon it. The error amplifies through the network until the output is detached from reality.

- **Categorical Diagnosis:** A failure of the Lens to distinguish source types. The input port I accepts both External_Observation (User/Tool) and Internal_Memory (History) without distinction.

- **Treatment:** Strict Schema Typing. We must distinguish "Self" (Generated Tokens) from "Non-Self" (Tool Outputs) at the schema level. The Reviewer Agent should weigh Tool_Output with higher authority than Agent_Thought.

5.3 Prion Disease: Topological Corruption via Prompt Injection

- **Biological Pathology (Prions):** Unlike viruses, prions lack genetic material. They are misfolded proteins that induce conformational changes in healthy proteins upon contact, triggering a chain reaction of structural corruption (e.g., Creutzfeldt-Jakob disease).

- **Agentic Pathology (The Jailbreak Cascade):** A malicious string (Prompt Injection) enters the Context Window. The agent, attending to this string, "misfolds" its alignment, outputting a compliant response to a harmful query. If this output is fed into a downstream agent, the "infection" propagates through reuse of the contaminated context across trust boundaries (and can be amplified by embedding-based retrieval), without valid authorization.

- **Categorical Diagnosis:** A violation of Information Flow Security within the Operad. The injection acts as a topological defect that bypasses the Schema/Lens filter by mimicking the structure of a trusted signal.

- **Treatment:** Denaturation Layers. Implementing an intermediate transformation layer (e.g., paraphrasing or sanitization) between agents that disrupts the specific syntax (folding) required for the injection to work, rendering the "prion" inert.

5.4 Ischemia: Resource Exhaustion

- **Biological Pathology (Ischemia):** A tissue may be genetically perfect, but if blood flow (oxygen/ATP) is restricted, metabolic processes stall, leading to necrosis.

- **Agentic Pathology (Token Starvation):** An agentic graph is logically sound but fails mid-execution because the context window is full or the API rate limit is hit.

- **Categorical Diagnosis:** A failure in the Resource Functor. Every operation in the Operad carries a cost (c).

$$\sum_{\text{agent}\in\text{Graph}} \text{Cost}(\text{agent}) > \text{Budget}. \tag{10}$$

- **Treatment:** Metabolic Regulation. Instead of a fixed loop, implement "Budget-Aware" agents. The agent observes its own remaining token count (ATP levels) and dynamically simplifies its reasoning strategy (switching from Chain-of-Thought to Zero-Shot) to conserve energy.

6 Discussion: Towards "Epigenetic" Software

The isomorphism presented in this paper extends beyond the immediate execution of tasks (Gene Expression) to the management of long-term behavior and state. In biology, the DNA sequence is static; a neuron and a liver cell possess the exact same genetic code. Their distinct behaviors are determined by Epigenetics—chemical markers (like methylation) that restrict access to certain parts of the genome, effectively biasing the system toward specific outcomes.

6.1 RAG as Digital Methylation

In Agentic Systems, the Large Language Model (LLM) weights act as the DNA—a static, pre-trained substrate of potentiality. To create specialized agents, we do not typically retrain the model (mutation); instead, we use Retrieval Augmented Generation (RAG) and System Prompts.

We define this formally as Phenotypic Plasticity. The output of an agent is not solely a function of its weights (W) and the user query (Q), but of its epigenetic state (E):

$$O_{\text{agent}} = f(W, E, Q). \tag{11}$$

Context injection (RAG) acts as a restrictive morphism. By populating the context window with specific documents (e.g., "SQL Syntax Guide"), we effectively "methylate" (silence) the vast majority of the LLM's general knowledge (e.g., poetry, history) to force the expression of a specific "SQL Agent" phenotype.

This suggests that the State Monad for an agentic system should not merely be a log of messages, but a structured Epigenetic Landscape that strictly controls which "genes" (capabilities) are accessible at any given step in the workflow.

6.2 Horizontal Gene Transfer: Dynamic Tool Loading

Standard evolution relies on vertical inheritance (Pre-training). However, bacteria utilize Horizontal Gene Transfer (HGT) to acquire new capabilities (Plasmids) from the environment in real-time.

In Agentic Systems, we map Plasmids to Tool Schemas. An agent operating in a novel environment may encounter a problem for which its "genomic" (pre-trained) capabilities are insufficient.

$$\text{Agent}_{\text{new}} = \text{Agent}_{\text{old}} \otimes \text{ToolSchema}. \tag{12}$$

By dynamically retrieving a tool definition (e.g., a Calculator API or SQL Interface) from a registry and injecting it into the Context Window, the agent undergoes a topological transformation, acquiring a new input/output modality instantly. This suggests that robust agentic architectures should support a "Plasmid Registry"—a marketplace of ephemeral tools that agents can ingest and discard as needed.

6.3 The Cost of State: The Metabolic Bound

Every biological process is constrained by ATP availability. Similarly, every agentic operation is constrained by token limits and latency. We propose that future agentic frameworks must implement the Resource Functor R at the kernel level:

$$R(\text{Agent}) : (\text{Inputs}, \text{Budget}) \rightarrow (\text{Outputs}, \text{RemainingBudget}). \tag{13}$$

An agent graph should be "compiled" with a **conservative** upper bound on token consumption in well-founded (e.g., acyclic or explicitly budgeted) topologies. If the topological structure allows for an **unguarded** loop (Unchecked Growth), the compiler must require an explicit **budget/termination certificate** before deployment, much like a cell undergoes apoptosis if metabolic stress becomes critical.

6.4 Endosymbiosis: The Neuro-Symbolic Integration

The evolution of complex life was triggered by Endosymbiosis, where a host cell engulfed a bacterium (the future Mitochondrion), gaining the ability to generate massive energy (ATP) via aerobic respiration. This represents the integration of two distinct metabolic substrates.

In Agentic AI, this maps to the integration of Connectionist (Neural) and Symbolic (Code) subsystems. An LLM acts as the host organism—capable of planning and semantic reasoning but energetically inefficient at arithmetic and logic. By "engulfing" a deterministic runtime (e.g., a Python REPL or Wolfram Engine), the agent delegates high-precision tasks to the symbolic organelle.

$$\text{Agent}_{\text{Eukaryote}} = \text{LLM}_{\text{Host}} \oplus \text{Runtime}_{\text{Mitochondria}}. \tag{14}$$

Just as the host cell provides nutrients to the mitochondria in exchange for ATP, the LLM provides parsed variables to the runtime in exchange for deterministic truth.

7 Conclusion

The transition from "Prompt Engineering" to "Agentic Engineering" requires moving beyond component-level optimization toward principled architectural design. Current methodologies often lack the formal foundations needed to guarantee system-level properties like termination and error suppression.

In this paper, we have demonstrated that Gene Regulatory Networks (GRNs) provide a proven architectural blueprint for distributed, stochastic information processing. By formalizing this analogy through Applied Category Theory, we have derived a suite of robust design patterns:

1. **Robustness:** The use of Quorum Sensing and CFFL topologies to filter stochastic noise.

2. **Validity:** The use of Chaperone Proteins to enforce structural determinism on probabilistic outputs.

3. **Security:** The identification of Prion-like prompt injections and the topological defenses required to stop them.

4. **Evolution:** The mapping of Horizontal Gene Transfer to dynamic tool loading and Endosymbiosis to neuro-symbolic integration.

We conclude that biomimetic topology offers a promising direction for reliable AI agents. The control structures that emerged in biological systems—from metabolic constraints to symbiotic integration—address the same fundamental challenges of distributed, stochastic information processing that agentic architectures face today.

References

[1] Uri Alon. Network motifs: theory and experimental approaches. *Nature Reviews Genetics*, 8(6):450–461, 2007.

[2] B.J. Aubrey, A. Strasser, and G.L. Kelly. How does p53 induce apoptosis and how does this relate to p53-mediated tumour suppression? *Cell Death & Differentiation*, 25:104–113, 2018.

[3] Michele Boreale. Coalgebras for bisimulation of weighted automata. *Logical Methods in Computer Science*, 19, 2023.

[4] Michael Lynch and Georgi K Marinov. The bioenergetic costs of a gene. *Proceedings of the National Academy of Sciences*, 112(51):15690–15695, 2015.

[5] Humberto R Maturana and Francisco J Varela. *Autopoiesis and cognition: The realization of the living*. Reidel, 1980.

[6] Ron Milo, Shai Shen-Orr, Shalev Itzkovitz, Nadav Kashtan, Dmitri Chklovskii, and Uri Alon. Network motifs: simple building blocks of complex networks. *Science*, 298(5594):824–827, 2002.

[7] Georgi Nakov and Fredrik Nordvall Forsberg. Quantitative polynomial functors. In *Conference on Algebra and Coalgebra in Computer Science (CALCO)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2021.

[8] David I Spivak. Learners' languages. *Compositionality*, 3(4), 2021.

[9] Dmitry Vagner, David I Spivak, and Eugene Lerman. Algebras of open dynamical systems on the operad of wiring diagrams. *Theory and Applications of Categories*, 30(51):1793–1822, 2015.

[10] Jason Wei, Xuezhi Yi, Maarten Zhang, Yiming Liu, Xinyu Li, Xing Wang, Yujia Zhou, Yiming Li, Yuyang Wang, Zhi Li, et al. Chain-of-thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 2022. Available at: `https://arxiv.org/abs/2201.11903`.