Spherepop

Geometry, Cognition, and the Transparency of Computation

Flyxion

2026

Typeset in TeX Gyre Pagella with LuaLaTeX.

Contents

Introduction: The Crisis of Flattened Thought

This introduction diagnoses the cognitive pathology that motivates the entire monograph: the systematic flattening of structured thought into opaque symbolic strings. It argues that modern computational and mathematical notation, despite its expressive power, routinely conceals the very structure it encodes, producing a kind of epistemic opacity that is neither necessary nor inevitable. Spherepop is introduced as a philosophical and computational response to this crisis.

Symbolic Compression and the Loss of Structure

Modern notation systems compress evaluation history into terminal outputs. This is not an accident of history, nor a correctable deficiency in any particular notational tradition. It is a design choice, made consistently and for intelligible reasons across centuries of mathematical and computational practice: the choice to optimize representational density at the cost of operational transparency. The choice produces notation that is concise, portable, and formally precise. It also produces notation that is, in a specific and diagnosable sense, cognitively

hostile — notation that requires its users to reconstruct, from surface syntax alone, the structural information that the notation has systematically suppressed.

The suppression is not total. A practiced reader of the expression $1 + 3 \times 2^2$ knows, without conscious effort, that the exponentiation must be resolved before the multiplication, and the multiplication before the addition. The knowledge is real. But it is implicit, encoded in a memorized convention rather than visible in the expression itself. The expression does not show its evaluation order; it presupposes that the reader will supply it from prior training. For a reader who has internalized the convention, this presupposition is harmless. For a reader who has not, the expression is structurally opaque: its surface form gives no indication of which operations are nested within which, which must be resolved first, or what the computation's trajectory through its intermediate states looks like. The terminal value 13 is accessible; the shape of the computation that produces it is not.

This monograph is a sustained investigation of that suppression: what it costs, why it is not necessary, and what a representational practice that recovers visible structure looks like. The investigation proceeds through eight parts that address the same underlying question from the directions of computation, cognition, thermodynamics, emergence, philosophy of science, semantics, geometry, and ethics. What holds the investigation together is a single structural claim: that the suppression of evaluation history is not merely a notational inconvenience but a systematic epistemological failure, and that recovering visible structure is not merely a pedagogical improvement but a philosophical necessity for genuine intelligibility across all

these domains.

Why Traditional Notation Becomes Cognitively Opaque

The cognitive opacity of conventional notation is not a mysterious phenomenon. It has a clear structural explanation: conventional notation compresses structural relations that are operationally meaningful into syntactic conventions that must be actively decoded. Operator precedence conventions compress evaluation order. Linear text compresses spatial nesting. Terminal values compress computation histories. In each case, the compression is lossy in a specific way: it preserves the extensional content — the correct answer — while discarding the intensional content — the structure of the process that generated it.

The cognitive cost of this compression is borne by the reader, who must reconstruct the discarded structure from the surface notation at the time of reading. This reconstruction is not free. It consumes working memory resources that would otherwise be available for understanding, consumes attentional resources that would otherwise support retention, and introduces error at exactly the points where the conventions are most complex — precisely the points where the structural information being suppressed is most important. A student who must simultaneously parse a nested expression, maintain a model of operator precedence, and track the current evaluation state is doing three cognitively demanding things at once, all because the notation has externalized none of them.

The alternative is not to eliminate compression — compres-

sion is necessary for cognition to function at all, as the analysis of Part II will establish — but to compress in ways that preserve the structural relations that matter for understanding. A notation system that makes evaluation order visible in its spatial structure does not require the reader to reconstruct evaluation order from memorized conventions. A notation system that encodes scope in physical containment does not require the reader to maintain an internal scope model from delimiter counting. The cognitive resources saved by these externalizations are not trivial. They are the difference between a notation system that supports understanding and one that merely encodes correct answers.

Computation as History Rather Than Instantaneous Result

The deepest claim of this monograph is not about notation. It is about the ontology of computation itself. The claim is that a computation is not primarily a mapping from input to output — a function that takes an expression and returns a value — but primarily a history: an ordered sequence of admissible transformation events through which a structured possibility space is navigated from an initial state to a terminal state. The terminal value is one feature of this history, and an important one. But it is not the whole of the computation, any more than the destination of a journey is the whole of the journey.

This claim has consequences that run through every subsequent chapter. If computation is fundamentally historical, then a representational system that preserves only terminal values is not merely incomplete but is systematically misleading

about what computation is. It presents the residue of a computation as though it were the computation itself, the endpoint of a trajectory as though it were the trajectory. Understanding a computation, on this view, requires access to the history — the ordered sequence of evaluation events, the constraint conditions in force at each step, the admissibility structure of the transformations performed — not merely to the terminal result. A pedagogy that delivers only terminal results trains students to produce correct answers without understanding why those answers are correct. A software system that exposes only terminal outputs provides no basis for understanding why those outputs were produced or whether they would remain correct under variations in the input.

Spherepop is the name given, in this monograph, to the formalism and the philosophical framework that takes this claim seriously. The name reflects the core operation: a *pop* event is the explicit, visible resolution of a locally admissible evaluation region — a bubble — into its result, with the resolution recorded in the computation's history rather than silently incorporated into a running total. The bubble is not merely a graphical convenience. It is a formal object: an admissibility region with a boundary that encodes scope, a history that records prior reductions, and a constraint structure that determines which further reductions are admissible. The pop event is not merely a computation step. It is a strongly admissible reduction that reorganizes the bubble's topology, preserves a transportable invariant, and modifies the future admissibility conditions of the computation in a way that the history records and the terminal value does not.

The Calculator as a Metaphor for Modern Abstraction

The handheld calculator is a nearly perfect embodiment of the epistemological condition this monograph sets out to examine. It accepts a structured expression as input and returns a terminal value as output, collapsing the entire computational history into an instantaneous result. The collapse is efficient: for the user who wants the answer and nothing else, the calculator is an ideal tool. But the collapse is also complete: no information about how the answer was reached is preserved, no record of the intermediate states is accessible, no indication is given of which operations were performed in which order or under which constraints. The computation is invisible. Only its residue is delivered.

This invisibility is not a limitation of the calculator's hardware. It is a design commitment: the decision that the terminal value is what matters and that the computational history is implementation detail. For most uses of calculators, this commitment is perfectly reasonable. For a student learning arithmetic, it is pedagogically destructive. The student who keys in $1 + 3 \times 2^2$ and receives 13 has learned that 13 is correct, but has learned nothing about why it is correct — why the exponentiation was resolved before the multiplication, what would have happened if the additions had been performed first, what the nested structure of the expression means and why it matters. The calculator has delivered the endpoint of an educational journey without supplying the journey itself.

The calculator is a metaphor — but only a metaphor — for a much broader condition in modern technical systems. Statis-

tical models, compiled software, opaque algorithmic decision systems, and large language models all share the calculator's fundamental architecture: they accept inputs and produce outputs, and the process mediating the two is invisible to the user and often to the system's designers. The scale is different. The consequences are different. But the structural situation is the same: a computation whose history has been suppressed, delivering a terminal value whose provenance is inaccessible. The critique this monograph develops applies to all of them, not by treating them as the same as calculators, but by identifying the common structural feature — the suppression of operational history — that makes them all instances of the same epistemological failure.

A Different Intuition: Geometry Before Symbolism

There is a different way of representing computation, one that is older than symbolic algebra and more deeply rooted in human spatial cognition than any formal notation system. It is the way of containers: the intuition that complex things are made of simpler things inside boundaries, that what is inside a boundary is local to it, that things inside smaller containers must be dealt with before the larger containers can be resolved. This intuition is not a mathematical theory. It is a cognitive primitive — one of the earliest and most neurologically robust organizational principles available to human minds.

Children understand containers before they understand formal syntax. They understand that things inside boxes are not the same as things outside boxes, that a box within a box is

at a different level from the box that contains it, that to get at the innermost object you must open the outermost container first. These are spatial and operational intuitions that develop in infancy, long before any formal mathematical training begins. Conventional mathematical notation does not exploit them. It compresses the nested structure of computation into a linear string of symbols and then requires the reader to reconstruct the nesting from precedence conventions that have no spatial correlate. The container intuition is available; the notation ignores it.

Spherepop exploits it. The nested bubble structure of Spherepop notation is not an arbitrary visual choice. It is a deliberate alignment between the formal structure of computation — which is genuinely hierarchical, genuinely nested, genuinely organized by containment and boundary — and the cognitive architecture of the human mind, which is uniquely well-equipped to process exactly that kind of spatial hierarchical organization. The alignment is productive not because it makes computation easier in some superficial sense, but because it makes the structural features of computation visible rather than hidden: the scope of each subexpression is physically enclosed, the evaluation order is geometrically encoded in nesting depth, the dependency structure is spatially apparent rather than syntactically implicit.

Spatial Cognition and Nested Scope

The spatial encoding of computational structure in Spherepop is not a pedagogical accommodation to visual learners. It is a consequence of a deeper principle about the relationship

between spatial cognition and structural understanding. Human brains process spatial hierarchies — nested containment, bounded regions, levels of enclosure — using cognitive machinery that is both developmentally early and neurologically robust. The same machinery that allows a child to understand that a toy is inside a box, which is inside a room, which is inside a house, is the machinery that Spherepop recruits for the understanding that an exponentiation is inside a multiplication context, which is inside an addition context. The structural operation is the same. The cognitive resources engaged are the same. Only the domain differs.

The significance of this is not merely that Spherepop notation is easier to read. It is that the cognitive resources recruited by spatial hierarchical processing are available in parallel with the symbolic processing resources recruited by conventional notation, distributing the cognitive load of comprehension across more of the brain's architecture. A reader processing a Spherepop diagram does not have to choose between perceiving the spatial structure and reasoning about the symbolic content. The spatial structure is the symbolic content: the containment relations encode the scope relations, the nesting depth encodes the evaluation priority, the bubble boundary encodes the scope limit. Cognitive geometry and symbolic computation are aligned rather than in competition.

Why Children Understand Containers Before Formal Syntax

The developmental priority of container understanding over formal syntax is well established in the cognitive science liter-

ature. Infants in the first year of life demonstrate sensitivity to containment relations: they distinguish objects that are inside containers from objects that are beside them, and they respond differently to violations of containment expectations. The understanding of nested containment — that one container can be inside another — develops in the second year. The understanding of hierarchical nesting — that multiple levels of containment form an ordered structure — develops across the preschool years, well before formal mathematical instruction begins.

Formal mathematical syntax, by contrast, is a late acquisition. The understanding of operator precedence conventions requires years of formal instruction and is imperfectly acquired even by many adults who use mathematical notation regularly. The conventions are arbitrary in the sense that they could have been different — nothing in the mathematical objects themselves requires that multiplication bind more tightly than addition — and they must be memorized as rules rather than perceived as structural features. They do not exploit the container intuition; they override it. The student who has a strong container intuition and encounters $1 + 3 \times 2^2$ will naturally read the expression left to right, performing the addition before the multiplication — exactly the order that the notation's conventions forbid but that the container intuition predicts. The precedence rules must then be learned as explicit overrides of a natural cognitive tendency.

Spherepop inverts this relationship. In Spherepop notation, the evaluation order is encoded in the containment structure: the innermost bubble is evaluated first not because of a memorized convention but because it is innermost, and container

intuition dictates that inner containers are resolved before outer ones. The student's natural cognitive tendency — to process inner containers before outer ones — is aligned with the correct evaluation order rather than in conflict with it. The cognitive work of learning Spherepop is not the work of overriding a natural tendency but the work of recognizing that a natural tendency correctly tracks a real structural feature of the computation.

The Central Thesis of the Essay

The central thesis of this monograph can now be stated precisely. Modern notation systems — mathematical, computational, and increasingly artificial — systematically optimize representational density at the cost of operational transparency. They compress evaluation histories into terminal values, suppress structural information that is operationally meaningful, and require users to reconstruct from surface syntax what has been hidden in the compression. This suppression is not accidental. It is a consequence of the historical conditions under which formal notation developed: conditions of paper, ink, typesetting, and linear textual transmission that rewarded conciseness and penalized spatial complexity. Those conditions no longer obtain in the same way. But the notational habits formed under them persist, and with them the epistemological costs.

Spherepop is a response to those costs. It is a theory of visible structure: a formalism, a cognitive model, and a philosophical framework that take seriously the claim that computation is fundamentally historical, that meaning is fundamentally navigational, and that intelligibility requires the preservation of

structural provenance across transformations. Against opacity, not against structure: the target of Spherepop's critique is not abstraction or formalism as such, but abstraction that conceals the operational geometry of what it abstracts. The proposal is not to abandon compression but to compress in ways that preserve what matters for understanding. The measure of what matters is the admissibility structure of the computation: the constraints that govern which transformations are possible at each step, which histories are admissible, and which terminal values are reachable by which routes.

Part I

Spherepop and the Geometry of Computation

Chapter 1

The Origins and Architecture of Spherepop

This chapter traces the origins of the Spherepop formalism and establishes its foundational architecture. Arithmetic is recast as nested containment, and the pop *operator is introduced as an explicit, visible evaluation event. The chapter argues that preserving computational provenance is not a luxury but a structural necessity for genuine understanding. It closes with a section of foundational definitions that stabilizes the monograph's core terminology: the primary register of Spherepop, the three levels of geometry, constraint as primitive, the ontology of a Spherepop object, layered admissibility, the Principle of Contextual Equivalence, understanding as stabilized navigational compression, and the scope of the monograph's physical claims.*

1.1 Arithmetic as Nested Containment

The claim that arithmetic can be represented as nested containment is not a metaphor. It is a structural observation about what

arithmetic expressions actually are, prior to any decision about how to notate them. An arithmetic expression like $1 + 3 \times 2^2$ is not a linear sequence of symbols. It is a hierarchically structured dependency graph in which certain subexpressions must be evaluated before others because they provide inputs to the operations that contain them. The exponentiation 2^2 must be evaluated before the multiplication $3 \times (\cdot)$ because the multiplication requires the result of the exponentiation as one of its operands. The multiplication must be evaluated before the addition $1 + (\cdot)$ for the same reason. The dependency structure is hierarchical, and the hierarchy is one of containment: the exponentiation is contained within the multiplication context, the multiplication is contained within the addition context.

Standard algebraic notation represents this hierarchy by a combination of operator symbols, grouping parentheses, and precedence conventions. The hierarchy is encoded, but it is not visible: a reader must actively reconstruct it from the surface syntax. Spherepop represents the same hierarchy by literal spatial containment: the exponentiation bubble is physically enclosed within the multiplication bubble, which is physically enclosed within the addition bubble. The reconstruction is not needed. The hierarchy is visible in the diagram's spatial structure, and the evaluation order — inner bubbles before outer bubbles — is a consequence of the spatial organization rather than a memorized convention imposed on top of it.

Construction 1.1 (Spherepop Representation of Arithmetic)**.** The Spherepop representation of an arithmetic expression is obtained by the following procedure:

(1) Identify the top-level operation — the operation whose result is the value of the entire expression. Draw an outer

bubble enclosing the entire expression.

(2) For each operand of the top-level operation that is itself a compound expression, draw an inner bubble enclosing that operand.

(3) Repeat recursively for each inner bubble until all bubbles contain only atomic values (numerals or variables).

(4) Label each bubble with its operation.

The resulting structure is a system of nested bubbles in which containment directly encodes evaluation dependency: a bubble B_1 is enclosed within a bubble B_2 if and only if the result of the computation within B_1 is required as an input to the computation within B_2, and B_1 must therefore be popped before B_2 can be popped.

For the expression $1 + 3 \times 2^2$, this procedure produces three nested bubbles: the innermost bubble enclosing 2^2 (labeled with the exponentiation operation), enclosed within a middle bubble enclosing $3 \times \mathsf{pop}(2^2)$ (labeled with multiplication), enclosed within an outer bubble enclosing $1 + \mathsf{pop}(3 \times \mathsf{pop}(2^2))$ (labeled with addition). The evaluation order — exponentiation, then multiplication, then addition — is immediately visible as the order of nesting: innermost first. No precedence convention is needed. The structural fact that the expression encodes is visible in the structure of the representation.

1.2 Pop as an Explicit Evaluation Event

The *pop* operation is the core primitive of the Spherepop formalism. A pop event is the evaluation of an innermost bubble — a bubble that contains no other bubbles — resulting in the replacement of that bubble by the value it computed. The key

features of the pop operation that distinguish it from ordinary algebraic reduction are visibility and historicity: the pop event is explicit rather than implicit, and it is recorded in the computation's history rather than silently incorporated into the expression's running value.

Definition 1.2 (Pop Event)**.** A *pop event* $\mathsf{pop}(B)$ at step t of a Spherepop computation is the locally admissible reduction of an innermost bubble $B = (U, \partial U, \sigma, \tau)$, where U is the internal content of the bubble, ∂U is its boundary, σ is its scope context, and τ is its accumulated evaluation history. The pop event:

(1) Evaluates the contents of U according to the operation labeling B, producing a result value v.
(2) Removes B from the current bubble topology $\mathcal{B}_t$, producing the updated topology $\mathcal{B}_{t+1} = \mathcal{B}_t \setminus \{B\}$.
(3) Substitutes v for the bubble B in the containing bubble, updating the containing bubble's content.
(4) Records the event (B, v, t) in the computation's global history H, preserving the bubble's identity, the value produced, and the step at which the pop occurred.

The pop event is locally admissible — it constitutes a strongly admissible reduction in the sense of Section 1.5 — if and only if B is an innermost bubble (no admissible bubble is nested strictly within B) and the constraint conditions on B's boundary ∂U are satisfied by the current evaluation state.

The explicitness of the pop operation — the fact that it is a named, recorded event rather than a silent transformation step — is not a formalism for its own sake. It reflects the ontological commitment that evaluation steps are history-forming events, not merely state-updating operations. A computation that has performed a pop is not in the same condition as a computation

that has not, even if the current expression state were (counterfactually) the same: the histories differ, and differing histories imply differing future admissibility conditions, by the path sensitivity principle of Section 1.5. The pop record is the medium through which history propagates through the computation.

1.3 Preserving Computational Provenance

Computational provenance is the record of how a result was reached: the ordered sequence of evaluation events, the constraint conditions in force at each step, the bubbles that were popped in which order, and the admissibility structure that governed each pop. Provenance is not the same as the result. Two computations with identical results may have entirely different provenances, and their difference in provenance is semantically significant under any collapse operator that preserves more than terminal values.

The preservation of computational provenance is the central practical commitment of Spherepop, and the argument for it has two levels. The first level is pedagogical: a student who has access to the provenance of a computation can understand why the result is what it is, can identify where errors entered the computation, and can reconstruct the reasoning that produced the result. A student who has access only to the result can verify it against an expected answer but cannot understand it, cannot diagnose errors in it, and cannot generalize from it to new problems with different structures. Provenance is the difference between a worked example and a black box.

The second level is formal: provenance is required for the correct assessment of semantic identity under the Principle of

Contextual Equivalence. Two computations that produce the same terminal value but via different provenances are identified by some collapse operators and distinguished by others. The choice of collapse operator is not arbitrary; it depends on what structural invariants are relevant to the use to which the result will be put. For a use that requires only the numerical value, the collapse that identifies them is appropriate. For a use that requires knowing the evaluation order — for instance, a pedagogical use, an audit use, or a use in a system where the computational cost of different evaluation paths matters — the collapse must be coarser, preserving the provenance rather than quotienting it away. The appropriate collapse depends on the context, and the context cannot be assessed without knowing what information the provenance contains. Erasing provenance forecloses the possibility of making this assessment correctly.

Remark 1.3. The formal relationship between provenance preservation and the action functional of Chapter 12 is direct: the action $S[\gamma] = \sum_t \mathcal{L}_t$ is defined over the full provenance γ of a computation, not over its terminal state. The distinction between high-action and low-action trajectories — between computationally costly and computationally efficient evaluation paths — is meaningful only for computations whose provenances are preserved. A system that discards provenance cannot measure computational cost in the action-functional sense; it can only verify that a terminal value was produced.

1.4 From Parentheses to Topology

The transition from parentheses to topology is a transition in the ontological status of the structural elements of computation. In standard algebraic notation, parentheses are syntactic markers: they instruct the reader to treat their contents as a unit for the purpose of precedence resolution, but they have no further formal status. They are erased in the process of evaluation: the expression (2^2) becomes 4, and the parentheses disappear. The structural information they encoded — that this subexpression was to be evaluated as a unit, before the operations that contained it — is consumed in the act of evaluation and does not survive into the result.

In Spherepop, the bubble that corresponds to a parenthesized subexpression is not a syntactic marker but a topological object: a bounded region of the computation's admissibility manifold, with a boundary that encodes constraint conditions, an interior that encodes the current state of the subcomputation, and a history that records the sequence of prior reductions within it. The bubble is not erased in evaluation; it is popped — transformed from an active evaluation region into a recorded event in the computation's history. The structural information it encoded — the scope of the subexpression, the constraints that governed its evaluation, the sequence of steps that produced its result — is preserved in the pop record rather than consumed in the act of evaluation.

This transition from erasure to preservation is not merely a notational change. It reflects the deeper ontological shift described in the Introduction: from treating computation as a mapping from input to output, to treating computation as an ordered history of admissible transformations through a struc-

tured possibility space. Parentheses are erased because, in the mapping conception, they are implementation detail — scaffolding that is needed during evaluation but has no standing in the final result. Bubbles are popped and recorded because, in the historical conception, each evaluation step is a permanent event in the computation's history, with consequences for the future admissibility structure that cannot be recovered from the terminal result alone.

The topological character of bubbles also has formal consequences that parentheses lack. Two parenthesized expressions can be syntactically identical while having entirely different semantic roles depending on context. Two bubbles with the same interior content but different boundary conditions — different scope contexts, different constraint configurations, different evaluation histories — are formally distinct objects with potentially different admissibility structures. The topology distinguishes what the syntax does not. This distinction matters wherever the structural context of a computation is as important as its terminal result, which, the arguments of this monograph maintain, is nearly everywhere.

1.5 Foundational Commitments and Stable Terminology

The material developed in this monograph spans pedagogy, cognitive science, thermodynamics, topology, philosophy of science, and the theory of computation. Such breadth invites terminological drift: the same word acquiring subtly different meanings in different chapters, the same concept appearing under different names in different domains, analogies hard-

ening imperceptibly into identity claims. The present section is designed to prevent that drift. It states, explicitly and in one place, the framework's core ontological commitments, its central distinctions, and the precise senses in which its key terms are used. Readers are encouraged to return to it when later chapters use terms in ways that might otherwise seem ambiguous.

The Primary Register of Spherepop

Spherepop is not primarily a notation system, nor primarily a formal calculus, nor primarily a philosophy of science. It is a theory of visible structure: a systematic account of why constrained systems — physical, cognitive, computational, semantic — become intelligible through the preservation of navigable relational structure across transformations, and become unintelligible through its erasure. The notation, the operators, the geometric models, the action formalism, and the philosophical arguments developed in this monograph are not separate inventions that happen to share terminology. They are local expressions of a single underlying claim:

> *Meaning, understanding, and computation arise through navigable transformations within constrained spaces of possibility. Preserving visible structure across those transformations is essential for genuine intelligibility.*

This claim operates simultaneously as a pedagogical principle (visible structure reduces cognitive load), a semantic principle (meaning is relational and navigational, not atomistic), a computational principle (evaluation histories contain information that terminal values do not), a geometric principle

(nested scope and dependency are more naturally expressed spatially than linearly), and a philosophical principle (modern technical systems increasingly optimize prediction at the cost of intelligibility and provenance). These registers are mutually reinforcing. The pedagogical principle is grounded in the cognitive science of working memory and semantic grouping. The semantic principle is grounded in the formal theory of admissible trajectories through constraint manifolds. The computational principle is grounded in the action formalism of Chapter 12. The geometric principle is grounded in the neuroscience of spatial and symbolic cognition examined in Part II. The philosophical principle is grounded in the political and ethical analysis of opacity developed in Part VIII. No register is primary; each provides evidence for the others.

Three Kinds of Geometry

The monograph makes extensive use of geometric language: admissibility manifolds, semantic configuration spaces, constraint fields, curvature in semantic space, topological scope, and so forth. It is essential that these uses be distinguished carefully, because they operate at three different levels.

Representational geometry is geometry as visualization aid: the use of spatial diagrams, nested bubbles, and topological pictures to make structural relations easier to perceive and remember. This is the most conservative claim, requiring no commitment beyond the empirical observation that spatial representations often support comprehension better than linear ones.

Cognitive geometry is the stronger claim that spatial cognition is structurally prior to symbolic cognition in human in-

telligence: that containment, nesting, locality, trajectory, and boundary are among the earliest and most neurologically robust cognitive structures, and that notation systems grounded in these structures therefore recruit cognitive resources that purely symbolic systems leave unused. This claim is empirical and is defended in Part II by appeal to the neuroscience of program comprehension, working memory, and semantic grouping.

Physical geometry is the claim that relational constraint structure is intrinsic to physical law: that physics is not merely describable geometrically but is organized through constraint manifolds, admissible trajectories, symmetries, and variational principles in a sense that is literal rather than metaphorical. This claim is defended in Part III in the thermodynamic domain, where it is made literally, and extended by structural analogy in Parts VII and VIII.

Spherepop's central geometric claim is that these three levels are deeply coupled through shared organizational principles — structural recurrence across scales — without being identical. The reason cognitive geometry works so well is not arbitrary: biological intelligence evolved to navigate a world that is itself geometrically organized at the physical level. The reason physical geometry is so productive in thermodynamics is not a coincidence: constraint structure is the common language through which both physical and cognitive systems become navigable. Geometry is therefore treated throughout this monograph as *the recurring structural language through which constrained systems become intelligible*, at whatever level the system operates.

Constraint as Primitive

The word *constraint* appears throughout the monograph in several apparently different senses: logical restriction, energetic restriction, semantic admissibility, computational tractability, topological obstruction, and evaluative boundary. These are not analogies for each other, nor are they accidentally sharing a name. The claim of this monograph is that they are all manifestations of one deeper structure: *constraint is the primitive*.

Definition 1.4. A *constraint* on a system X is any condition that restricts the set of admissible transitions from the current state of X. The *admissibility set* $\mathsf{Adm}(C)$ of a constraint configuration C is the subset of all states accessible from the current state that satisfy all constraints in C. Constraints may be logical (valid inference rules), energetic (conservation laws), semantic (coherent interpretation conditions), computational (type constraints, scope rules), or topological (boundary conditions on evaluation regions). All are instances of the same formal structure: a restriction of the admissibility set.

The unification of these senses under a single primitive is not a casual metaphorical move. It is the foundational commitment that makes the monograph's cross-domain arguments possible. When Part III argues that thermodynamic entropy is accessibility volume and that Spherepop's collapse operator is structurally identical to thermodynamic coarse-graining, the argument depends on this unification. When Part II argues that working memory limitations and scope rules are both constraint structures governing cognitive navigation, the argument depends on it. When Part VI argues that semantic grounding is a constraint satisfaction problem and that opacity is a failure

of constraint visibility, the argument depends on it. Readers who resist the unification will find the cross-domain arguments unconvincing. Readers who accept it will find them convergent.

The Ontology of a Spherepop Object

A Spherepop object is not a state, a value, a symbol, or an expression. It is *an admissible historical trajectory through a structured possibility space*. This is the monograph's primary ontological commitment, and it has consequences that run through every subsequent chapter.

The ontology is layered, and the layers are non-arbitrary:

Region. A structured possibility space defining which transformations are admissible. Regions are the primary geometric objects: they constitute the admissibility manifold $\mathcal{A}(X)$ of a computation, the phase space of a thermodynamic system, the semantic configuration space of an interpretation. Regions are primary *geometrically*.

History. An ordered trajectory $\gamma = (e_1, e_2, \ldots, e_n)$ through a region, recording the sequence of admissible transitions actually traversed. Histories are the primary *semantic* objects: they carry the provenance, the evaluation order, the constraint sequence, and the commitment profile that terminal values do not contain. Histories are primary semantically.

Process. The local dynamics that generate individual transitions along a trajectory: the mechanism by which each step e_t is produced from the current state and constraint configuration. Processes are primary *dynamically*, but they are not the most fundamental objects, because the same process may generate different histories under different constraint configurations.

Collapse. The formation of equivalence classes by identifying

histories that preserve specified structural invariants under a chosen quotient map $\mathsf{collapse}(q)$. Collapse is an operation on histories, not a primitive. Its output — an equivalence class of histories — is a derived object, not a foundational one.

This ordering matters philosophically. In conventional computation, states are primary and histories are implementation detail. In Spherepop, histories are primary and states are compressed summaries of navigational histories through structured possibility spaces. The terminal value 13 produced by the evaluation of $1 + 3 \times 2^2$ is not the computation; it is the residue of the computation after its history has been discarded. The computation is the ordered sequence of bubble pops, with their associated constraint configurations, commitment costs, and admissibility records.

Layered Admissibility

Admissibility in Spherepop operates at two distinct levels, and conflating them is a source of potential confusion.

Local admissibility is binary: a single evaluation step e_t either satisfies the constraint conditions in force at step t or it does not. There is no intermediate grade. Local admissibility defines the boundary of the admissibility manifold: inadmissible steps lie outside it, admissible steps lie within.

Definition 1.5. $\mathsf{Adm}_{\text{local}}(e_t \mid C_t) \in \{0, 1\}$: the step e_t is locally admissible relative to constraint configuration C_t if and only if all active constraint conditions in C_t are satisfied by the transition e_t effects.

Global admissibility is graded: a complete trajectory γ through the admissibility manifold has a real-valued quality measure

determined by the action functional:

$$A(\gamma) = F(S[\gamma],\ \Omega[\gamma],\ C[\gamma]),$$

where $S[\gamma]$ is the total action cost, $\Omega[\gamma]$ is the remaining accessibility volume at the trajectory's terminal state, and $C[\gamma]$ is the total structural commitment accumulated along the trajectory. Among all locally admissible trajectories, global admissibility distinguishes more from less optimal ones.

The philosophical significance of this layered structure is expressed most compactly by the phrase: *local admissibility defines the manifold; action defines its curvature*. Inadmissible trajectories do not exist within the manifold. Among admissible trajectories, action determines the geometry: which are geodesics, which are costly, which are structurally stable under variation. This is the computational analogue of the distinction between possibility and preference: constraints determine what is possible; action determines what is preferred among the possible.

The Principle of Contextual Equivalence

The identity conditions of Spherepop objects are not absolute. Two histories γ_1 and γ_2 are semantically equivalent only relative to a collapse operator that preserves the invariants relevant to the current admissibility structure:

$$\gamma_1 \sim_q \gamma_2 \quad \Longleftrightarrow \quad \mathsf{collapse}(q)\,(\gamma_1) = \mathsf{collapse}(q)\,(\gamma_2).$$

Different collapse operators preserve different structural invariants and therefore induce different equivalence relations. Two programs that produce identical outputs may be identified under a collapse operator that preserves only terminal results,

while remaining distinct under a collapse operator that also preserves the admissibility profile, the commitment trajectory, or the cognitive legibility of the evaluation sequence.

Axiom 1.6 (Contextual Equivalence)**.** Semantic identity in Spherepop is always relative to a collapse operator. There is no context-independent notion of semantic equivalence that is not implicitly indexed to a choice of which structural invariants to preserve. The question is never "are these two histories the same?" but always "are these two histories the same under this collapse?"

This principle is the formal expression of the monograph's repeated critique of premature collapse: the identification of histories whose structural differences are semantically significant under the relevant admissibility conditions. The calculator that returns 13 has collapsed $1+3\times2^2$ under a collapse operator that preserves only numerical value. Whether that collapse is semantically valid depends entirely on what one needs from the computation. For a student learning operator precedence, it is not valid: the evaluation history is precisely the information the student needs, and its collapse is obliteration, not compression.

Understanding as Stabilized Navigational Compression

Understanding, within the Spherepop framework, is not a static possession or a mental state. It is a structural property of a cognitive system's relationship to an admissibility manifold, constituted by three successive and mutually reinforcing phases:

Navigation is the first phase: the ability to traverse the admissibility manifold correctly, predicting valid transitions, avoiding inadmissible regions, and preserving relevant constraints at

each step. A student who can correctly evaluate $1 + 3 \times 2^2$ by following the evaluation order without knowing why it is correct has achieved navigational competence without full understanding.

Stabilization is the second phase: the ability to navigate correctly across variations in framing, representation, scale, and context. A trajectory that succeeds in one exact form but fails under minor perturbation is fragile. True understanding survives perturbation: changes in notation, examples, domain, or level of abstraction do not destroy navigational competence. Stabilization is invariant-preserving navigation under transformation — the analogue, in the cognitive domain, of a trajectory that is stationary under variations in the action formalism.

Structured compression is the third phase: the formation of reusable invariant-preserving models that support continued admissible navigation across new contexts. Compression alone is not understanding; a lookup table is compressed but unintelligent. Understanding occurs when compression preserves the structural relations — the operational geometry of the domain — that are necessary for navigation in situations the original training did not include.

Definition 1.7. Understanding of a domain D is the possession of a compressed model of D's admissibility manifold that supports stable navigation across variation in framing and context. Formally, a cognitive system S understands D to the extent that S's internal model $\widehat{\mathcal{A}}_S(D)$ is a structure-preserving compression of the actual admissibility manifold $\mathcal{A}(D)$, stable under the class of perturbations relevant to D.

This definition grounds the monograph's central distinction between simplification and obliteration in the theory of under-

standing: simplification is structure-preserving compression of the admissibility manifold; obliteration is compression that destroys structural relations needed for navigation. A student who has been given only the answer 13 has been given an obliterated model. A student who has watched the sequence of bubble pops and internalized the evaluation order as a navigational skill has been given a simplified model that preserves the operational geometry of arithmetic.

The Scope of Physical Claims

The physical claims in this monograph are of two different kinds, and they must not be conflated.

In thermodynamics, the claims are made *literally*. Entropy is, formally and not merely metaphorically, the logarithm of the accessibility volume of the admissibility set. Thermodynamic evolution is, literally, constraint navigation. The increase of entropy in isolated systems is, literally, the expansion of the accessibility volume as internal constraints relax. The Spherepop collapse operator is, formally, a coarse-graining map whose effect on accessibility volume is captured by the formula $S_q = S - k_B \ln |\ker q|$. These are not analogies to physics; they are physics, stated in the language of the Spherepop framework.

In other domains — cognition, semantics, computation, pedagogy — the claims are made by *structural analogy*. The analogy is not casual: it tracks genuine shared organizational principles, and the argument of the monograph is that this sharing is not accidental but reflects structural recurrence across scales. But the claim is not that cognitive processing is literally thermodynamic, or that semantic interpretation is literally a physical process. The claim is that the same structural pat-

tern — constraint navigation through admissibility manifolds with action-governed trajectory selection — recurs across these domains because it is a sufficiently general organizational principle to appear wherever systems must navigate structured possibility spaces under constraint.

The distinction can be stated precisely: thermodynamic claims in this monograph are made in the mode of *literal application*; cognitive, semantic, and computational claims are made in the mode of *structural isomorphism*. The isomorphism is argued, not assumed, and the argument is that systems as different as thermodynamic ensembles, programming languages, and human cognition independently converge on the same organizational architecture because that architecture is the most general form that navigable constraint satisfaction can take.

The Universal Admissibility Schema

The central formal object of the Spherepop framework is the *universal admissibility schema* $\mathfrak{S}$, a six-tuple that parameterizes the abstract structure of any strongly admissible reduction:

Definition 1.8 (Universal Admissibility Schema)**.** A *universal admissibility schema* is a tuple

$$\mathfrak{S} = (B,\ \delta,\ \mathcal{C},\ H,\ R,\ \mathcal{I}),$$

where:

- B is the *admissibility region* — a bounded space of locally possible continuations, equipped with a constraint membrane ∂B;
- δ is the *perturbation* — an incoming evaluation event, input, or contextual signal seeking admission through ∂B;

- $\mathcal{C}$ is the *constraint structure* — the active set of conditions governing which perturbations the membrane admits;
- H is the *historical state* — the accumulated record of prior reductions that actively shapes current admissibility conditions;
- R is the *reduction operator* — the map $R : (B, \delta, \mathcal{C}, H) \to (B', H')$ that produces a new admissibility region and an updated history when a compatible perturbation is admitted; and
- $\mathcal{J}$ is the *transportable invariant* — the structural identity that must satisfy $\mathcal{J}(B) \cong \mathcal{J}(B')$ for the reduction to be strongly admissible.

The local compatibility condition is written $\delta \vdash_{\mathcal{C}} B$. A perturbation that fails this test is refused; the region does not reorganize, and the history records the refusal. A perturbation that passes the test triggers the reduction R, producing B' and H'.

The schema $\mathfrak{S}$ is the monograph's true spine. Every subsequent chapter develops a specific domain by providing concrete realizations of the six components. The realizations are not analogies to each other; they are parameterizations of the same abstract reduction topology under different semantic interpretations. The table below gives the principal instantiations:

Domain	B	δ
Lambda calculus	lambda term	argument
Thermodynamics	macrostate	constraint relaxation
Cognition	semantic configuration	percept/proposition
Biology (stem cell)	epigenetic state	chemical signal
Spherepop (arithmetic)	bubble topology	pop event
AI inference	latent semantic manifold	token/context update

The power of the schema lies not in the breadth of the table but in the precision of the mapping: for each domain, one must identify exactly what plays the role of the admissibility region, what plays the role of the constraint membrane, what the transportable invariant is, and what failure of invariant transportability looks like in that domain. The schema is a research grammar: it forces the analyst to make explicit the admissibility structure that is typically left implicit in domain-specific accounts.

State Space and History Space

Standard computation theory is *state-primary*: it treats the current state of a computation as the fundamental object, and the computation's history as implementation detail — a record of how the state was reached, useful for debugging but not constitutive of the computation's meaning. On the state-primary view, two computations that pass through the same state at the same step are identical at that step, regardless of how they reached that state.

Spherepop is *history-primary*. The fundamental distinction is formal: let X denote the *state manifold* — the space of possible instantaneous states of a computation — and let $\Gamma(X)$ denote the *history space* — the space of all admissible ordered sequences of transitions through X:

$$\Gamma(X) \;=\; \{\, \gamma = (x_0, e_1, x_1, e_2, x_2, \ldots, e_n, x_n) \;\mid\; \text{each } (x_{k-1}, e_k, x_k) \text{ is locally adr}$$

The state manifold X is the quotient of $\Gamma(X)$ under the collapse that identifies all histories sharing the same terminal state: $X \cong \Gamma(X)/{\sim_v}$, where $\gamma_1 \sim_v \gamma_2$ iff γ_1 and γ_2 terminate at the same state. The state is derived from the history, not the other way around.

Collapse operators act over $\Gamma(X)$, not over X:

$$\mathsf{collapse}(q) : \Gamma(X) \;\longrightarrow\; \Gamma(X)/{\sim_q}.$$

Different collapse operators produce different quotients of the history space, preserving different structural invariants. The state-primary view corresponds to the single collapse $\mathsf{collapse}(v)$ that discards all history beyond the terminal state. The Spherepop framework insists that this collapse is one option among many, appropriate in some contexts and destructive in others. The ontological primacy of histories — the claim that $\Gamma(X)$ is the fundamental object and X is a derived quotient — is the formal expression of the monograph's deepest commitment: that *the trajectory is the object*.

Trajectory Energetics

The variational structure of Spherepop — introduced at length in Chapter 12 — is not an addition to the framework but a con-

sequence of the history-primary ontology. Once histories are primary objects, it is natural to ask which histories are preferred among those that are locally admissible: which trajectories through $\Gamma(X)$ the system actually realizes, and what structural property selects them. The answer is the action functional.

For any admissible history $\gamma \in \Gamma(X)$, the *action* is:

$$S[\gamma] = \int_0^T L(\gamma(t),\ \dot{\gamma}(t),\ C(t))\, dt,$$

where L is the Lagrangian — a real-valued function of the current state, the rate of state change, and the active constraint configuration — and T is the total duration of the trajectory. In the discrete setting appropriate to Spherepop computation, the integral becomes the sum $S[\gamma] = \sum_t \mathcal{L}_t$ developed in Chapter 12. The global admissibility score of a trajectory is:

$$A(\gamma) = F(S[\gamma],\ \Omega[\gamma],\ C[\gamma],\ H[\gamma]),$$

where $\Omega[\gamma]$ measures the remaining accessibility volume at the trajectory's terminal state, $C[\gamma]$ measures accumulated structural commitment, and $H[\gamma]$ encodes historical dependence. The preferred trajectories — the geodesics of the admissibility manifold — are the stationary points of $S[\gamma]$ under admissible variation, precisely as in classical mechanics.

This variational structure unifies the framework's treatment of thermodynamics (entropy as accessibility volume, second law as trajectory selection toward high-Ω terminal states), cognition (understanding as stabilization of low-action semantic trajectories), computation (optimal evaluation order as the geodesic through the semantic configuration space), and AI training (learning as the selection of model parameters that

minimize action over the training-history manifold). In each case, the action functional is not a metaphor borrowed from physics; it is the natural measure over the history space that the history-primary ontology demands.

Global Admissibility and the Hallucination Signature

The distinction between local and global admissibility, introduced in the layered admissibility framework above, requires a formal apparatus for measuring the degree to which locally admissible reductions fail to assemble into globally coherent structures. The appropriate apparatus is sheaf cohomology.

Let $\mathcal{A}$ be the admissibility manifold of a computation, and let $\{U_i\}$ be an open cover of $\mathcal{A}$ by local admissibility regions. For each U_i, let $\mathcal{F}(U_i)$ denote the set of locally admissible reduction sequences over U_i. The assignment $U_i \mapsto \mathcal{F}(U_i)$ defines a sheaf $\mathcal{F}$ over $\mathcal{A}$, with restriction maps that describe how local reduction sequences restrict to smaller regions. A global section of $\mathcal{F}$ is a globally admissible trajectory: a reduction sequence that is locally admissible on every patch and that glues consistently across all overlaps.

Definition 1.9 (Global Admissibility)**.** A history $\gamma \in \Gamma(X)$ is *globally admissible* if and only if its local admissibility patches $\{\gamma|_{U_i}\}$ admit a coherent global section of the sheaf $\mathcal{F}$. The obstruction to global admissibility is measured by the first Čech cohomology group $\check{H}^1(\mathcal{A}, \mathcal{F})$. A history is globally inadmissible — subject to a gluing obstruction — if and only if $\check{H}^1 \neq 0$.

The condition $\check{H}^1 \neq 0$ is the *hallucination signature*: the formal, domain-independent measure of the failure to assemble locally coherent patches into a globally admissible structure. It

unifies the following phenomena under a single mathematical condition:

An AI language model that generates locally fluent text but globally asserts impossible facts has $\check{H}^1 \neq 0$ in its semantic sheaf: the local consistency of each sentence does not propagate to a globally consistent narrative. A biological lineage reconstruction that assigns plausible local transitions but produces a globally impossible developmental tree has $\check{H}^1 \neq 0$ in its lineage sheaf. A physical theory that is locally consistent in each domain of application but fails to compose into a unified description has $\check{H}^1 \neq 0$ in its correspondence sheaf. In each case, the formal condition is the same: local admissibility is necessary but not sufficient for global coherence, and the measure of the gap between them is the first cohomology of the appropriate sheaf.

The Admissibility Recurrence Theorem

The culminating formal result of the Foundational Definitions section is the theorem that makes the monograph's cross-domain arguments structurally unavoidable rather than merely plausible:

Theorem 1.10 (Admissibility Recurrence)**.** *Let* $\mathfrak{S} = (B, \delta, \mathcal{C}, H, R, \mathcal{J})$ *be any system that satisfies the four conditions of Strong Admissibility: local compatibility, recursive restructuring, invariant transportability, and historical modification of future admissibility. Then:*

(*i*) *The system exhibits path sensitivity:* $H_1 \neq H_2 \Rightarrow \mathsf{Adm}(B \mid H_1) \neq \mathsf{Adm}(B \mid H_2)$.

(*ii*) *The system's history space* Γ *supports a well-defined action functional* $S : \Gamma \to \mathbb{R}$.

(*iii*) *The globally admissible trajectories are the stationary points of*

S under admissible variation.

(*iv*) *The obstruction to global admissibility is measured by* $\check{H}^1$ *of the sheaf of locally admissible reductions.*

(*v*) *These structural properties hold independently of the specific substrate realization of* $\mathfrak{S}$*.*

Proof sketch. (i) follows directly from condition (4) of Strong Admissibility: if the history genuinely modifies future admissibility conditions, then distinct histories produce distinct admissibility structures, establishing path sensitivity. (ii) follows from the history-primary ontology: Γ is a well-defined space, and any function $S : \Gamma \to \mathbb{R}$ that satisfies the locality and additivity conditions of the Lagrangian formalism is an action functional. (iii) is the principle of stationary action applied to Γ: the Euler-Lagrange equations for S determine the locally optimal trajectories, which are the geodesics of the admissibility manifold. (iv) is the definition of the gluing obstruction in the sheaf $\mathcal{F}$ of locally admissible reductions: a globally coherent trajectory exists if and only if the local patches glue, and the obstruction to gluing is $\check{H}^1$. (v) follows from the schema-level generality of conditions (i)–(iv): they are derived from the abstract conditions of $\mathfrak{S}$ without reference to any specific substrate. □

The Admissibility Recurrence Theorem establishes that the structural properties Spherepop formalizes — path sensitivity, action-functional trajectory selection, and sheaf-cohomological obstruction to global coherence — are not special features of Spherepop computation. They are consequences of the universal admissibility schema, and they therefore hold in any domain that instantiates the schema. The later chapters of this monograph are demonstrations of this recurrence: not metaphorical

extensions of a computational idea, but formal verifications that thermodynamics, cognition, biology, and physics each instantiate $\mathfrak{S}$ with appropriate parameters, and thereby inherit the full structure of the theorem.

Strong Admissibility and Path Sensitivity

Not every locally compatible reduction constitutes a genuinely admissible collapse in the full sense the framework requires. A reduction is *strongly admissible* if and only if it satisfies four conditions simultaneously:

Definition 1.11 (Strong Admissibility)**.** A reduction $R(B_i, \delta_i) \to B_{i+1}$ is strongly admissible if and only if:

(1) **Local compatibility:** $\delta_i \vdash_{C_i} B_i$ — the perturbation satisfies the active constraint structure of the region.

(2) **Recursive restructuring:** the region B_{i+1} is not merely a filtered version of B_i but undergoes genuine internal reorganization — its topology, constraint structure, or evaluation geometry is modified by the absorbed perturbation.

(3) **Invariant transportability:** there exists a structural invariant $\mathcal{I}$ such that $\mathcal{I}(B_i) \cong \mathcal{I}(B_{i+1})$ — a meaningful identity survives the collapse in a form that remains usable in future reductions.

(4) **Historical modification of future admissibility:** the updated history H_{i+1} genuinely constrains future admissibility conditions — $\mathsf{Adm}(B_{i+1} \mid H_{i+1}) \neq \mathsf{Adm}(B_{i+1} \mid H_i)$ — meaning the reduction has altered what the region can absorb next.

Condition (4) is the deepest. It formalizes what distinguishes Spherepop from Markovian computation: the requirement that the history of a system is not merely a record but an

active determinant of future admissibility. This is *path sensitivity*:

$$H_1 \neq H_2 \;\Rightarrow\; \mathsf{Adm}(B \mid H_1) \neq \mathsf{Adm}(B \mid H_2).$$

Two systems in identical present states but with different histories possess different future admissibility structures. Their current snapshots are extensionally equivalent; their admissibility manifolds are not. This is why the evaluation history of $1+3\times2^2$ carries information the terminal value 13 does not: the history determines which further operations on the result are admissible in the current context, which collapses have already been committed, and which possibilities have been foreclosed. The terminal value forgets all of this. The history preserves it.

Path sensitivity also explains why condition (2) — recursive restructuring — is necessary and why its absence is diagnostic of a non-instance of genuine admissibility collapse. A system that merely filters outputs through a fixed rule without reorganizing its internal topology in response to its history is not performing strongly admissible reduction; it is performing lookup. The thermostat is the paradigm case: it satisfies condition (1) in every cycle, but never satisfies conditions (2), (3), or (4). Its topology does not change, no invariant survives as a usable structural identity, and its admissibility conditions are identical regardless of how many prior reductions it has performed.

Degenerative Collapse and the Gluing Obstruction

The failure taxonomy of Spherepop is as important as its positive conditions. A reduction may be locally compatible — condition (1) satisfied — while failing to constitute a strongly admissible collapse. The framework identifies two principal

failure modes.

Definition 1.12 (Degenerative Collapse)**.** A collapse $R(B_i, \delta_i) \to B_{i+1}$ is *degenerative* if and only if the recursive restructuring it performs destroys the transportability of invariant structure — formally, if no invariant $\mathcal{I}$ exists such that $\mathcal{I}(B_i) \cong \mathcal{I}(B_{i+1})$ in a form usable by subsequent reductions. Degenerative collapse produces a new region B_{i+1} that cannot participate in further strongly admissible reductions because the historical and structural continuity required for condition (3) has been severed.

A shattered glass is the physical paradigm of degenerative collapse: the perturbation (impact) satisfies local compatibility (the glass can be struck), and a violent reorganization occurs, but no transportable invariant survives. The shards cannot function as an admissibility region for further meaningful reductions. The collapse is real but degenerative. The calculator's erasure of evaluation history is a computational instance of the same failure: the reduction satisfies condition (1) and produces an output, but the historical continuity required for conditions (3) and (4) is destroyed. The result cannot be used to reconstruct what the computation was, only what it produced.

The second failure mode arises not at individual reduction steps but at the global level of assembling locally admissible reductions into a coherent structure:

Definition 1.13 (Gluing Obstruction)**.** A gluing obstruction occurs when a collection of locally strongly admissible reductions $\{R_i\}$ over overlapping admissibility regions $\{U_i\}$ cannot be assembled into a globally coherent reduction structure, because the compatibility conditions on the overlaps $U_i \cap U_j$ are

violated. Formally, a gluing obstruction is present when the first cohomology $H^1 \neq 0$ in the sheaf of locally admissible reductions over the admissibility manifold.

The gluing obstruction unifies several apparently distinct phenomena. An AI language model that produces locally grammatical and semantically coherent sentences but globally hallucinates a factually impossible narrative has satisfied local admissibility at every step while incurring a gluing obstruction at the global level: the local patches cannot be assembled into a globally coherent history. A biological lineage reconstruction model that assigns plausible local developmental transitions but produces a globally impossible developmental tree has the same failure structure. In both cases, local coherence does not guarantee global coherence, and the measure of the failure is the obstruction $H^1 \neq 0$: the presence of a twist in the data that prevents the local patches from being glued into a globally admissible surface.

The gluing obstruction is the formal counterpart, in the language of Spherepop, of the distinction between locally admissible and globally admissible trajectories introduced in the layered admissibility framework above. A trajectory may satisfy local admissibility at every step while accumulating a global incoherence that makes the full history strongly inadmissible as a unit. Spherepop's insistence on preserving the full evaluation history is precisely the insistence on maintaining the information needed to detect and diagnose such global failures — information that is irretrievably lost when evaluation histories are collapsed to terminal values.

Three Levels of Correspondence

The Admissibility Recurrence Theorem demonstrates structural properties that hold across domains, but it does not assert that those domains are the same substance or that their admissibility structures are identical. The framework's claims operate at different levels of strength, and clarity about those levels prevents the most serious misreadings.

Structural analogy is the weakest claim: two domains share a similar reduction grammar — a family resemblance between their admissibility structures — without the correspondence being formally precise. Structural analogy is a heuristic that motivates investigation but does not constitute a proof. When this monograph observes that the Feynman path integral and the Spherepop history operator have a similar philosophical character, the claim is at the level of structural analogy.

Operational isomorphism is the central claim of the framework: two domains instantiate the same abstract admissibility schema $\mathfrak{S}$ with different parameters, producing reduction topologies that are formally isomorphic at the level of the universal admissibility grammar. When this monograph argues that thermodynamic coarse-graining and the Spherepop collapse operator satisfy the same factorization condition and the same entropy-reduction formula, the claim is at the level of operational isomorphism. This is the level at which the Admissibility Recurrence Theorem operates, and it is the strongest claim the framework makes across distinct physical domains.

Physical identity is the claim that two domains share the same substrate dynamics: not merely the same abstract structure, but the same physical mechanism. This monograph makes physical-identity claims only within thermodynamics, where

the literal identification of entropy with accessibility volume and of constraint relaxation with spontaneous evolution is supported by statistical mechanics. It does not make physical-identity claims across domains — cognition is not literally thermodynamic, and computation is not literally quantum mechanical — and readers should treat any language that appears to make such claims as shorthand for the weaker operational isomorphism.

The framework's intellectual ambition lies entirely at the level of operational isomorphism: the demonstration that the same abstract reduction topology — the universal admissibility schema with path sensitivity, action-functional trajectory selection, and sheaf-cohomological obstruction — recurs across domains that appear superficially unrelated. That recurrence is the content of the Admissibility Recurrence Theorem, and it is what the subsequent chapters demonstrate.

Chapter 2

Operational Semantics Through Geometry

Locality, evaluation regions, scope as physical boundary, and dependency as spatial nesting are developed here as the operational semantics of Spherepop. The chapter culminates in a formal treatment of refusal, collapse, and admissibility as geometric conditions on evaluation.

2.1 Locality and Evaluation Regions

The operational semantics of Spherepop is organized around a single fundamental principle: *locality*. Every evaluation event in a Spherepop computation is local to the bubble in which it occurs: it affects only the interior of that bubble, it is governed only by the constraint conditions on that bubble's boundary, and its consequences propagate outward only through the recorded result that replaces the popped bubble in the containing expression. Nothing that happens inside a bubble can affect the contents of another bubble directly; it can affect them only

by changing the value that the popped bubble contributes to its container.

This locality principle is not merely a design choice. It reflects the structure of admissibility itself. The admissibility conditions on a bubble's boundary define what is possible within that bubble: what evaluation steps are admissible, what constraints apply, what results can be produced. The admissibility conditions are local to the bubble: they are determined by the bubble's scope context and the constraint configuration of its enclosing bubbles, not by the contents of sibling bubbles at the same nesting level. Two bubbles at the same level within a containing bubble may be evaluated in either order — neither is constrained by the state of the other — precisely because neither's admissibility conditions depend on the other's interior.

Definition 2.1 (Evaluation Region)**.** An *evaluation region* is a bubble $B = (U, \partial U, \sigma, \tau)$ together with the set of locally admissible evaluation steps that can be performed within it: the set of pop events $\mathsf{pop}(B')$ for innermost bubbles $B' \subseteq U$, bind events $\mathsf{bind}_{a<b}(B)$ that apply additional constraints to B's interior, and refuse events $\mathsf{refuse}(B')$ that record the inadmissibility of a potential pop. The evaluation region is the formal correlate of the intuitive notion of a computation's local context: the bounded space within which a specific subcomputation takes place, with its specific admissibility conditions and its specific history.

The locality of evaluation regions has a consequence that is central to the Spherepop framework's treatment of concurrency and evaluation order independence. Because evaluation events within one bubble cannot directly affect the contents of sibling bubbles, the relative order in which sibling bubbles are popped is semantically immaterial: the result of the containing bubble

is the same regardless of whether the left sibling or the right sibling is popped first. This is the Spherepop analogue of the confluence theorem for reduction systems: different evaluation orders produce the same terminal result, because the result of each bubble is determined only by its own interior content and boundary conditions, which are unaffected by the evaluation of siblings.

2.2 Scope as Physical Boundary

The boundary of a bubble is its scope. Everything inside the boundary is within the scope of the operations and constraints defined by that bubble's evaluation region. Everything outside is not, and cannot be directly affected by what happens within. The boundary is not merely a notational marker; it is a physical constraint on the propagation of evaluation effects. Pop events, refuse events, and bind events that occur within a bubble cannot cross its boundary to affect the contents of the containing bubble except through the single channel of the pop result: the value that replaces the bubble when it is popped.

This physical conception of scope as boundary has several consequences that differ importantly from the standard treatment of scope in programming language theory. In standard presentations, scope is a syntactic property of declarations: a variable x declared within a block has scope limited to that block, meaning that references to x outside the block are not resolved to the same declaration. The scope rule is a rule about name resolution, enforced at the level of the language's binding semantics. It is a property of the textual program rather than a physical feature of an evaluation structure.

In Spherepop, scope is a physical property of the evaluation geometry. The boundary of a bubble is the physical limit of the effects of operations within it. Variables introduced by bind events within a bubble have scope limited to that bubble not because a name-resolution rule forbids their use outside it, but because the bind event modifies the constraint configuration of the bubble's interior, and that modification cannot cross the bubble's boundary except through the pop result. The scope rule is a consequence of the locality of evaluation, not an independent constraint imposed on top of it.

Proposition 2.2 (Scope-Locality Correspondence)**.** *In the Spherepop formalism, the scope of any bind event* $\mathsf{bind}_{a<b}(B)$ *is precisely the interior* U *of the bubble* B *to which it is applied. No operation outside* B *can observe the constraint introduced by the bind event except through the pop result of* B*, which reflects the constraint only to the extent that the constraint affected the value produced by the evaluation of* U*.*

The physical conception of scope as boundary also makes scope errors visible in a way that syntactic scope rules do not. In a textual program, a scope error — a reference to a variable outside its scope — is a violation of a textual rule that is not immediately apparent from reading the code. In a Spherepop diagram, a scope error would require a reference line to cross a bubble boundary, which is visually apparent as a structural violation. The diagram makes the error visible rather than requiring a separate static analysis to detect it.

2.3 Dependency as Spatial Nesting

The dependency structure of a computation — which subcomputations must be completed before which others can begin — is encoded in Spherepop by spatial nesting. A bubble B_1 nested within a bubble B_2 depends on B_1: the computation within B_2 cannot proceed past the point where it requires the result of B_1 until B_1 has been popped. The nesting relation is the dependency relation, expressed geometrically.

This encoding of dependency as nesting has a formal correlate in the partial order on pop events imposed by the bubble topology. If bubble B_1 is nested within bubble B_2, then the pop event $\mathsf{pop}(B_1)$ must precede the pop event $\mathsf{pop}(B_2)$ in any admissible evaluation sequence: $\mathsf{pop}(B_1) \prec \mathsf{pop}(B_2)$. The partial order is determined entirely by the nesting structure of the bubble topology, without any need for explicit dependency annotations or topological sort algorithms. The geometry encodes the order; the geometry is the semantics.

The expressiveness of nesting as a dependency encoding is limited in one important respect: it can represent only *tree-structured* dependency relations — relations in which each bubble has at most one containing bubble. In computations with shared subexpressions — where the result of one subcomputation is used in multiple places — the dependency relation is a directed acyclic graph rather than a tree, and cannot be represented by pure nesting without duplication of the shared subexpression. The Spherepop framework handles this through the bind operation, which can introduce a constraint that references the result of a prior pop event, effectively creating a dependency arc that crosses the nesting hierarchy. This is a generalization of the variable-binding mechanism of functional programming

languages, expressed in the geometric language of the Spherepop formalism.

2.4 Refusal, Collapse, and Admissibility

The *refuse* operator $\mathsf{refuse}(B)$ is the formal record of a locally inadmissible reduction: the record that a potential pop event was considered and rejected because the constraint conditions on the bubble's boundary were not satisfied by the current evaluation state. The refuse operator is not a failure mode; it is a positive event in the computation's history, recording the fact that a specific constraint was operative at a specific step and prevented a specific evaluation.

The cognitive analogue is exact: a mind that refuses to make an inference it cannot justify is not failing to think; it is thinking correctly. The refuse event in Spherepop is the computational analogue of epistemically responsible deference: the acknowledgment that the current admissibility conditions do not support the proposed reduction, and that proceeding with it would violate the constraint structure of the ongoing computation. Systems that perform computations without refuse events — that always produce an output regardless of whether the admissibility conditions are satisfied — are systems that can hallucinate: systems that generate results without grounding them in the constraint structure that would make those results genuinely admissible.

The *collapse* operator $\mathsf{collapse}(q)$ is distinct from both the pop operator and the refuse operator. Where pop resolves a single innermost bubble into its result and records the event, collapse identifies two or more histories under an equivalence

relation $\sim_q$, quotienting the space of trajectories into equivalence classes of histories that preserve specified structural invariants. Collapse is an operation on the space of histories, not on individual bubbles. Its formal conditions — the requirement that the quotient map be admissibility-preserving, and the factorization condition $f = \bar{f} \circ \mathsf{collapse}(q)$ for observable functions — ensure that collapse is a structured compression rather than a destructive erasure. The collapse operator is the mechanism by which Spherepop achieves generalization: the identification of histories that are equivalent for the purposes of the current computation, while preserving the distinction of histories that differ in ways that are semantically significant.

Chapter 3

The Spherepop Operators

A systematic treatment of the core operators: $\mathsf{pop}(\cdot)$, $\mathsf{refuse}(\cdot)$, *collapse operators and quotient spaces, and bind operations and constraint formation. Histories and evaluation chains are shown to be the primary semantic objects, not instantaneous values.*

3.1 Histories and Evaluation Chains

A Spherepop computation history is a finite ordered sequence of evaluation events:

$$\mathcal{H} = (e_1, e_2, \ldots, e_n), \quad e_k \in \{\mathsf{pop}(B)\,,\ \mathsf{refuse}(B)\,,\ \mathsf{collapse}(q)\,,\ \mathsf{bind}_{a<b}(B)\},$$

where each event is admissible relative to the constraint configuration and bubble topology in force at the step of its occurrence. The history is the primary semantic object of the Spherepop formalism: it is the object over which the action functional $S[\mathcal{H}]$ is defined, the object whose equivalence classes are formed by collapse operations, and the object that carries the provenance information the terminal value discards.

An evaluation chain is a subsequence of a history consisting only of pop events and the bubbles they resolve. The evaluation chain of a computation traces the sequence of reductions that transformed the initial expression into its terminal value, recording which bubbles were popped in which order and what values they produced. The evaluation chain is the minimal subhistory required to reconstruct the terminal value from the initial expression; the full history may additionally contain refuse events and bind events that constrained the evaluation without themselves contributing directly to the terminal value.

Definition 3.1 (Evaluation Chain)**.** The *evaluation chain* of a Spherepop history $\mathcal{H} = (e_1, \ldots, e_n)$ is the subsequence

$$\mathcal{H}_{\text{eval}} = (e_{k_1}, e_{k_2}, \ldots, e_{k_m})$$

consisting of all and only the pop events in $\mathcal{H}$, in the order of their occurrence. The evaluation chain is the compressed projection of the full history under the collapse operator that identifies histories with identical pop-event sequences; it preserves the evaluation order and the results of each reduction while discarding the refuse and bind events.

The evaluation chain is an important intermediate object because it is the natural level at which many questions about computation are asked: questions about evaluation order, computational complexity, and result correctness are all questions about the evaluation chain. But the evaluation chain is not the full history, and the information lost in projecting to the evaluation chain — the refuse events that recorded constraint violations, the bind events that introduced scope constraints — is genuine semantic information that matters for questions

about admissibility, provenance, and the relationship between the computation and its context.

3.2 The Meaning of $\mathsf{pop}(\cdot)$

The pop operator, introduced informally in Chapter 1 and defined formally in Section 1.2, is the primary evaluation primitive of the Spherepop formalism. Its meaning in the formal system is the following: given an innermost bubble B in the current bubble topology $\mathcal{B}_t$, the pop event $\mathsf{pop}(B)$ performs the locally admissible reduction of B, producing a result value v, updating the bubble topology to $\mathcal{B}_{t+1} = \mathcal{B}_t \setminus \{B\}$, and recording the event in the computation's history.

The pop operator is a strongly admissible reduction in the sense of the Foundational Definitions: it satisfies local compatibility (the bubble's boundary conditions are met), performs recursive restructuring (the containing bubble's content is updated to reflect the pop result), preserves a transportable invariant (the mathematical value computed by the bubble's operation), and modifies future admissibility conditions (by removing the popped bubble from the topology and potentially enabling the pop of the containing bubble, which was previously blocked by the presence of a non-popped inner bubble).

The interpretive content of the pop operator goes beyond its formal definition. To pop a bubble is to make a commitment: to convert a potential computation into an actual result, foreclosing the alternative paths that would have been available if the bubble had remained unresolved. In the variational framework of Chapter 12, this commitment has a cost — it reduces the accessibility volume of the remaining computation — and it

is irreversible in the sense that the popped bubble cannot be un-popped without reverting to a prior state of the computation's history. The pop operator is the computational analogue of an irreversible thermodynamic process: it reduces entropy locally (by resolving an undetermined expression to a specific value) while increasing the entropy of the computation's global history (by adding a new determination event to the record of past commitments).

3.3 The Meaning of refuse(·)

The refuse operator refuse(B) is the formal complement of the pop operator. Where pop commits to a reduction, refuse records the non-commitment: the fact that a potential reduction was considered and rejected because the local admissibility conditions were not satisfied. The refuse event is not a failure but a structural determination: the determination that the computation is not yet in a state where the proposed reduction is admissible.

The refuse operator has no analogue in conventional computation, where the absence of a reduction is simply the absence of a step, with no formal record and no propagation of information about why the step was not taken. In Spherepop, the refuse event is a first-class citizen of the history: it is recorded alongside pop events, it contributes to the computation's action cost (by preserving accessibility volume rather than reducing it, at the cost of deferred commitment), and it carries information about which constraints were operative at the step of the refusal.

The philosophical significance of the refuse operator is that

it formalizes *principled non-action*: the refusal to perform a computation that is formally possible but contextually inadmissible. A system without refuse operators performs every locally possible reduction, without regard to whether the constraint conditions are satisfied. Such a system is the computational analogue of a reasoner who draws every locally valid inference without checking whether the premises are established or the background conditions are met. The refuse operator is the mechanism by which Spherepop models the kind of constraint-sensitive deferral that is characteristic of careful reasoning: the willingness to not-pop a bubble that could be popped, because the constraint conditions for doing so admissibly are not currently in force.

3.4 Collapse Operators and Quotient Spaces

The collapse operator family $\{\mathsf{collapse}(q)\}$ is indexed by equivalence relations q on the space of admissible histories. Each collapse operator identifies histories that are equivalent under q, producing a quotient space of equivalence classes. The formal conditions on collapse operators ensure that the quotient is semantically coherent: the factorization condition guarantees that any observable function that is invariant under the equivalence descends uniquely through the collapse map, so that the collapse preserves everything that matters for the purposes of the chosen equivalence relation.

The most important special cases are:

The *terminal-value collapse* $\mathsf{collapse}(v)$ identifies all histories with the same terminal value, producing a quotient whose equivalence classes are the sets of histories that compute the

same result. This is the collapse implicitly performed by any system that discards computation history and returns only a final output.

The *evaluation-chain collapse* $\mathsf{collapse}(e)$ identifies histories with the same evaluation chain — the same sequence of pop events — while distinguishing histories that differ in their refuse or bind events. This is a finer collapse than the terminal-value collapse, preserving evaluation order while abstracting over constraint-management events.

The *admissibility-profile collapse* $\mathsf{collapse}(a)$ identifies histories with the same sequence of admissibility determinations — the same record of which reductions were admitted, which were refused, and under which constraint configurations — while abstracting over the specific values produced. This is the collapse relevant for auditing and verification: it preserves the structural record of the computation's constraint-compliance while abstracting over the specific results.

The *full-provenance collapse* $\mathsf{collapse}(0)$ is the identity: it identifies only identical histories with themselves, preserving all structural information. This is the collapse appropriate when full provenance is required, such as in forensic analysis of a computation or in educational settings where the evaluation process itself is the subject of study.

3.5 Bind Operations and Constraint Formation

The bind operator $\mathsf{bind}_{a<b}(B)$ introduces a new constraint into the evaluation region of bubble B: the constraint that value a is less than value b, or more generally, that a specified relation holds between specified quantities within the scope of B.

The bind event narrows the admissibility set of the evaluation region — it adds a constraint that rules out some previously admissible evaluation paths — and records the constraint introduction in the computation's history.

The bind operator is the Spherepop analogue of variable binding in programming languages and of constraint introduction in constraint satisfaction systems. Its formal effect is to modify the constraint configuration C of the bubble B: after the bind event $\mathsf{bind}_{a<b}(B)$, the constraint configuration becomes $C \cup \{a < b\}$, and any evaluation step within B that would violate this constraint is no longer locally admissible. The constraint is scoped to the bubble B — it affects only the evaluation of B's interior — and it is released when B is popped.

Chapter 4

Spherepop as a Cognitive Interface

This chapter argues that Spherepop functions as a cognitive prosthetic: it externalizes working memory, reduces hidden cognitive load, and enables visual computation with semantic transparency. A sustained comparison with flat string representations grounds the argument empirically.

4.1 Externalizing Working Memory

The human working memory system — the cognitive faculty responsible for maintaining and manipulating information during active cognitive tasks — has a limited capacity that constrains the complexity of computations that can be performed mentally. The limit is not fixed at a universal number of items; it depends on the nature of the items, the availability of chunking strategies, and the degree to which long-term memory structures can be recruited to support maintenance. But the limit is real, and it is routinely exceeded by the demands of under-

standing complex computations represented in conventional notation.

Spherepop reduces the working memory demands of computation comprehension by externalizing the primary sources of cognitive load. The scope model — the representation of which variables are in scope, which constraints are active, and which evaluation regions are currently open — is externalized into the visible bubble structure of the diagram: the scope is the bubble's interior, the active constraints are encoded in the bubble's boundary conditions, and the open evaluation regions are the bubbles that have not yet been popped. A reader does not need to maintain an internal scope model; the scope model is the diagram.

The evaluation order — the representation of which subexpressions must be evaluated before which others — is externalized into the nesting structure: inner bubbles before outer bubbles. The reader does not need to maintain an internal evaluation order representation or apply precedence rules to determine which operations to perform first; the order is visible in the spatial organization. The computation state — the representation of which subexpressions have been evaluated and which remain — is externalized into the pop history: popped bubbles become values, unpopped bubbles remain as bounded regions. The reader does not need to maintain a separate mental model of the computation's progress; the progress is visible in the diagram's current state.

4.2 Reducing Hidden Cognitive Load

The cognitive load imposed by conventional notation is largely hidden: it is not experienced as a distinct burden because it has been internalized through years of practice. An experienced mathematician reading $\sum_{k=1}^{n} k^2$ does not consciously experience the cognitive cost of recovering the summation structure from the notation; the recovery is automatic. But the cost is real, as demonstrated by the systematic errors that occur when the notation is sufficiently unfamiliar or the expression sufficiently complex to exceed the capacity of the automated processing. The hidden cognitive load surfaces as error under pressure.

Spherepop makes the cognitive load visible precisely by removing it: when the scope, evaluation order, and computation state are externalized into the diagram, the mental effort that was previously consumed by maintaining internal representations of these structural features becomes available for understanding the computation's content. The diagram does cognitive work that the reader's working memory would otherwise have to do, freeing those resources for higher-level comprehension.

This externalization is not merely a pedagogical convenience for beginners. Expert users of a notation system still bear the cognitive load imposed by that system, even when they bear it automatically. The automatic processing of conventional notation consumes attentional and working memory resources that are not available for simultaneous reasoning about the computation's implications. A notation system that requires less automatic processing to recover structural information leaves more resources available for the kind of reflective engagement with the computation that produces genuine understanding

rather than competent execution.

4.3 Visual Computation and Semantic Transparency

The phrase *visual computation* refers to computation in which the visual structure of the representation directly encodes the semantic structure of the computation: where what the diagram looks like is, in a precise sense, what the computation means. Spherepop achieves visual computation in this sense: the spatial containment structure of the bubble diagram directly encodes the dependency structure of the computation, the nesting depth directly encodes the evaluation priority, the bubble boundary directly encodes the scope limit. The diagram is not a picture of the computation; it is the computation, expressed in a medium that exploits human spatial cognition rather than working against it.

Semantic transparency is a property of a representation system: a representation is semantically transparent to the extent that its semantic content is directly readable from its surface form without reconstruction from implicit conventions. Conventional algebraic notation is semantically opaque: its semantic content — the evaluation order, the scope structure, the dependency relations — must be reconstructed from syntactic markers and memorized conventions. Spherepop notation is semantically transparent in the relevant respects: the evaluation order is visible in the nesting, the scope is visible in the containment, the dependency relations are visible in the inclusion structure.

Semantic transparency is not the same as simplicity. A complex nested Spherepop diagram may be visually complex. But

its visual complexity is a direct reflection of its semantic complexity: the diagram is complex because the computation is complex, and every feature of the diagram's complexity corresponds to a feature of the computation's structure. A complex conventional expression, by contrast, may be visually simple — a string of symbols and operators — while encoding a complex computation whose structure is invisible. The visual simplicity of the conventional expression is achieved by hiding the complexity, not by reducing it.

4.4 Why Flat Strings Are Hard to Think With

The flatness of conventional notation — its representation of hierarchically structured computations as linear sequences of symbols — is the source of its cognitive difficulty. Human cognition is not naturally flat. It processes hierarchical structures through spatial cognition that is organized around containment, nesting, and boundary, not through sequential symbol processing. The sequential symbol string of conventional notation is not the natural format for human cognition; it is a compromise between the spatial complexity of hierarchical representation and the linear constraints of written text on a page.

The sequential format requires the reader to perform a specific cognitive operation that Spherepop notation does not: the mental reconstruction of a hierarchical spatial structure from a linear symbolic sequence. This reconstruction is not trivial. It requires maintaining an internal representation of the current nesting depth, tracking the opening and closing of parentheses or the application of precedence rules, and updating the

representation at each symbol in the sequence. The reconstruction is error-prone when nesting is deep, when operators with different precedences are interleaved, or when the reader is simultaneously trying to understand the semantic content of the expression rather than merely parsing its syntactic structure.

The difficulty is not that flat strings are inherently impossible to understand. With sufficient practice, a reader can reconstruct the hierarchical structure from the flat string automatically. But the automaticity of the reconstruction is not the same as the absence of cognitive cost. The cognitive cost is paid in the form of resources devoted to reconstruction that are not available for understanding, and it accumulates as expressions become more complex. Spherepop eliminates this cost by providing the hierarchical structure directly, in a form that spatial cognition can process without reconstruction.

Chapter 5

Spherepop and Programming Languages

Spherepop is situated within the broader landscape of programming language theory. Parse trees versus spatial regions, context windows as topological neighborhoods, and the relation to functional programming are examined. The chapter argues that Spherepop's contribution is not merely aesthetic but operationally substantive.

5.1 Operational Semantics and Language Design

The operational semantics of a programming language is the formal specification of how programs in that language are executed: what the execution steps are, how they modify the program's state, and what the relationship is between the source-language expression and the sequence of machine-level operations that it produces. Conventional presentations of operational semantics describe reduction rules — rules that specify how one expression is transformed into another — and apply those rules sequentially until a value is reached. The evaluation

history is implicit in the sequence of rule applications, but it is not a first-class object of the semantics: it is a device for defining the meaning of expressions, not an object of study in its own right.

Spherepop's operational semantics makes the evaluation history a first-class object. The meaning of a Spherepop expression is not its terminal value but the set of strongly admissible histories that lead from the initial expression to terminal values: the admissibility manifold of the expression, equipped with the action functional that assigns costs to trajectories through it. The terminal value is a derived notion — the last entry in the evaluation chain of any history in the expression's admissibility manifold — rather than the primary semantic object.

This shift has consequences for language design. A language whose operational semantics takes histories as primary will naturally expose evaluation histories to programmers: it will provide mechanisms for inspecting, recording, and reasoning about the history of a running computation. It will distinguish between computations that produce the same output via different histories, because those computations have different semantic profiles in a history-sensitive semantics. It will provide collapse operators that allow the programmer to choose the appropriate level of history preservation for a given task.

5.2 Parse Trees Versus Spatial Regions

The parse tree of a programming language expression is the conventional data structure used to represent its hierarchical syntactic structure. Like the nested bubble structure of a Sphere-

pop diagram, a parse tree encodes the dependency relations among the expression's subexpressions: the parent-child relation in the tree corresponds to the containment relation in the Spherepop bubble structure. The two representations are structurally isomorphic as graphs.

The difference lies in what the representation makes visible and what it suppresses. A parse tree is a data structure that is typically invisible to the programmer: it is constructed internally by the parser and used to drive evaluation, but it is not exposed as a first-class object of the language. The programmer writes a flat source string; the compiler internally constructs the parse tree; the parse tree drives code generation or interpretation; the programmer never sees the tree. The hierarchical structure of the computation is present in the parse tree but invisible to the user of the language.

In Spherepop, the bubble structure — the analogue of the parse tree — is the primary interface. The programmer does not write a flat string and rely on a parser to recover the structure; the programmer draws or constructs the bubble structure directly, making the hierarchical dependency explicit in the representation rather than leaving it implicit in a flat string. The parse tree and the bubble structure are isomorphic as abstract structures, but they differ in their epistemic status: the parse tree is an internal implementation artifact; the bubble structure is the user-visible semantic representation.

5.3 Context Windows as Topological Neighborhoods

A context window in a programming language is the set of bindings, types, and constraints that are in scope at a given point in a program: the local semantic environment that determines what names refer to, what operations are admissible, and what types are expected. Context windows are typically implicit in programming language implementations — they are maintained internally by the type checker or interpreter — but they are one of the primary sources of cognitive difficulty for programmers, who must maintain a mental model of the current context window while reading code.

In Spherepop, the context window of an evaluation step is exactly the bubble that contains it: the interior of the current evaluation region, bounded by the bubble's boundary, with the constraint configuration defined by the active bind events within that region. The context window is not implicit; it is the visible bubble. Knowing the context window requires only reading the boundary conditions of the current bubble, which are explicit features of the Spherepop diagram.

The topological characterization of context windows as neighborhoods in the admissibility manifold is more than a metaphor. The context window at a given evaluation step is literally a neighborhood in the semantic configuration space of the computation: the set of evaluation states that are reachable from the current state by locally admissible transitions, without crossing the current bubble's boundary. Two evaluation steps are in the same topological neighborhood — the same context window — if and only if they are within the same bubble. The

bubble structure of the Spherepop diagram is a topological atlas for the computation's semantic configuration space: each bubble is a coordinate chart, and the nesting relations among bubbles define the gluing conditions that assemble the local charts into the global admissibility manifold.

5.4 The Relation to Functional Programming

Functional programming languages — languages in which computation is modeled as the application of functions to arguments, without mutable state or side effects — are the conventional programming paradigm closest in spirit to the Spherepop formalism. Both treat computation as the evaluation of expressions, both use hierarchical nesting to encode the order of operations, and both have well-developed theories of evaluation order and reduction strategies. Lambda calculus, the mathematical foundation of functional programming, is the canonical formal system for the study of expression reduction, and its beta-reduction rule is the prototype for the pop operator of Spherepop.

The differences between functional programming and Spherepop are not differences in underlying mathematical structure but differences in the objects that are treated as primary. In functional programming, values are primary: the meaning of an expression is its value, computation is the process of finding that value, and the evaluation history is an implementation detail. In Spherepop, histories are primary: the meaning of an expression is the admissibility manifold of its evaluation histories, values are derived from histories, and the evaluation history is a semantic object in its own right.

The relationship between Spherepop and functional programming is therefore the relationship between a history-sensitive and a value-sensitive semantics for the same underlying class of computations. For computations where only the terminal value matters, the two semantics agree: the Spherepop evaluation chain produces the same terminal value as the functional-programming evaluation. For computations where the evaluation history matters — for understanding, for verification, for audit, for pedagogy, for complexity analysis — the Spherepop semantics provides information that the functional-programming semantics discards. The Spherepop formalism is a conservative extension of functional-programming semantics: it adds history-sensitivity without changing the terminal-value behavior, at the cost of requiring richer representations and admitting a richer space of equivalence relations.

Part II

Cognition, Neuroscience, and Program Understanding

Chapter 6

The Neuroaesthetics of Programming

Program comprehension is treated as a semantic and neurological activity. Drawing on brain imaging studies of programming and cognitive science research on working memory, this chapter establishes why scope visualization is not merely pedagogically useful but reflects the architecture of human semantic cognition.

6.1 Program Comprehension as Semantic Activity

There is a persistent and largely unexamined assumption embedded in the way programming is taught and discussed: that writing and reading code is fundamentally a logical activity, that the programmer's cognitive task is essentially that of a formal reasoner manipulating symbols according to syntactic rules. On this assumption, the difficulty of programming is a difficulty of logic — of keeping track of complex inference chains, of applying rules correctly, of avoiding logical errors in reasoning about the behavior of abstract machines. The

assumption is natural enough: programming languages are formal systems, their semantics are defined by mathematical specifications, and correctness in programming is determined by logical criteria. But the assumption is empirically wrong, or at least seriously incomplete, in ways that have significant consequences for how programming environments should be designed and how programming should be taught.

The evidence against the purely logical model of programming cognition comes from several converging directions. First, from the phenomenology of programming itself: experienced programmers report that their understanding of code is organized around semantic units — functions, modules, patterns, design strategies — rather than around syntactic or logical structure. When an experienced programmer reads an unfamiliar codebase, they do not process it as a sequence of logical propositions to be evaluated; they recognize familiar patterns, assign semantic roles to components, construct a high-level model of the code's purpose, and then use that model to navigate the lower-level details selectively. The process is top-down and semantically organized, not bottom-up and syntactically driven.

Second, from the study of programming errors: the kinds of mistakes programmers make are not primarily the kinds of mistakes that a formal reasoner would make — incorrect application of inference rules, violations of logical constraints — but rather the kinds of mistakes that arise when a semantic model is incorrect or incomplete. Off-by-one errors, wrong assumptions about variable scope, incorrect mental models of data structure behavior: these are errors in the programmer's internal representation of the code's semantic content, not in

their ability to apply formal rules to syntactic expressions. The formal rules are generally known; the semantic model is where the difficulty resides.

Third, and most directly relevant to the concerns of this monograph, from the observation that the difficulty of reading code scales with its structural opacity rather than with its logical complexity. Code that is logically complex but structurally transparent — where the nesting, scope, and dependency relations among components are visually evident from the code's layout — is experienced as easier to understand than code that is logically simpler but structurally opaque, where the scope and dependency relations must be reconstructed from surface syntax using implicit conventions. This asymmetry points directly toward the central thesis of Spherepop: that the difficulty of program comprehension is largely a difficulty of recovering structural information that conventional notation has compressed away.

6.2 Working Memory and Attention

Working memory — the cognitive system responsible for maintaining and manipulating information over short time scales during the performance of a task — is a limited resource in human cognition. The classical characterization of working memory capacity as approximately seven plus or minus two items, proposed by Miller in 1956, has been refined by subsequent research to suggest that the effective limit is closer to three or four distinct chunks of information that can be actively maintained and manipulated simultaneously, with the exact capacity depending on the nature of the items, the degree of

chunking, and the availability of long-term memory structures to support the maintenance of information in a compressed form.

The relevance of working memory limitations to programming comprehension is direct and well-documented. When a programmer reads a piece of code, they must maintain in working memory a representation of the current state of the program — the values of relevant variables, the current scope, the constraints imposed by prior control-flow decisions — while simultaneously parsing the current expression and integrating it with the maintained representation. If the code's structural organization does not support the chunking of this information — if scope boundaries are not visually marked, if dependency relations between components must be inferred from non-local syntactic cues, if the evaluation order of nested expressions must be reconstructed from memorized precedence rules — then the working memory demands of comprehension increase sharply.

The specific bottleneck that Spherepop is designed to address is the cost of maintaining *implicit scope* in working memory. In standard linear notation, the scope of a subexpression — the set of other expressions on which it depends and by which it is constrained — is not directly visible. It must be inferred from syntactic structure, parenthesization, indentation conventions, and operator precedence rules. This inference itself consumes working memory resources, because the programmer must simultaneously maintain the current scope model and update it as they parse each new syntactic element. In Spherepop's nested bubble notation, scope is made physically explicit: the boundary of a bubble is the scope boundary of its

contents, and the nesting relation of bubbles directly encodes the dependency relation of the expressions they contain. The working memory cost of maintaining the scope model is reduced to near zero, because the scope model is externalized into the visible structure of the notation rather than maintained as an internal representation. [I This externalization of scope is not merely a convenience. It is a cognitive prosthetic in the precise sense: a representational structure that takes over a cognitive function that would otherwise have to be performed internally, at the cost of working memory resources, thereby freeing those resources for other aspects of the comprehension task. The analogy with other cognitive prosthetics — written language, which externalizes memory; mathematical notation, which externalizes calculation; diagrams, which externalize spatial reasoning — is direct. Spherepop externalizes scope tracking, which is one of the most cognitively demanding aspects of program comprehension.

6.3 Why Code Is Not Experienced as Pure Logic

The claim that code is experienced as semantic content rather than as formal structure is not merely a phenomenological observation. It has a neurological basis that has become increasingly clear as brain imaging methods have been applied to the study of programming comprehension. The relevant finding — which will be examined in more detail in the following section — is that reading and understanding code activates brain regions associated with semantic processing and working memory rather than, or in addition to, regions associated with mathematical reasoning. This finding is consistent with

the phenomenological evidence and with the error-pattern evidence: programming comprehension is a semantic activity, organized around meaningful units and their relations, rather than a purely logical activity organized around formal rules and their application.

The philosophical implication is significant. If code comprehension is fundamentally semantic, then the legibility of code is fundamentally a function of how well its surface form supports semantic interpretation — how clearly its meaningful units are delimited, how transparently its dependency and scope relations are expressed, how naturally its evaluation order is encoded in its visual structure. A notation system optimized for semantic legibility will look very different from one optimized for syntactic conciseness or formal precision. Standard mathematical and programming notation has been optimized primarily for conciseness and formal precision, under the implicit assumption that the reader will supply the semantic structure through internalized conventions. Spherepop is optimized for semantic legibility, under the explicit assumption that the notation should carry as much semantic structure as possible in its visible form.

The contrast becomes clearest when comparing two representations of the same computation. The expression $1 + 3 \times 2^2$ is formally precise and syntactically concise. A reader familiar with operator precedence conventions can recover its semantic structure — exponentiation first, then multiplication, then addition — from the surface syntax, but the recovery requires an active inference step that consumes working memory resources and is prone to error when the expression is more complex. The Spherepop representation, with its nested bubbles mark-

ing the evaluation regions explicitly, makes the same semantic structure directly visible. The evaluation order is not inferred from convention; it is read from the geometry of the nesting. A reader who has never encountered operator precedence conventions can read the structure from the diagram, because the diagram's spatial organization directly encodes the information that the linear notation encodes only implicitly through a compressed convention.

This is why code is not experienced as pure logic: because the cognitive task of reading code is not the application of logical rules to syntactic expressions, but the reconstruction of a semantic model from a compressed representation that encodes semantic structure implicitly. The difficulty of programming is largely the difficulty of this reconstruction, and the solution Spherepop proposes is not to make the logical rules clearer but to make the semantic structure explicit.

6.4 The Brain Imaging Studies of Programming

The application of functional magnetic resonance imaging to the study of programming comprehension has produced a body of evidence that substantially revises the naive picture of programming as purely logical activity. Studies in which participants are asked to understand code segments while their brain activity is recorded consistently reveal activation in regions associated with semantic processing — in particular, regions in the left inferior frontal gyrus and the temporal-parietal junction that are also activated during natural language comprehension and semantic memory retrieval — as well as regions associated with working memory maintenance, including the dorsolateral

prefrontal cortex and parietal areas involved in visuospatial attention.

The activation of language-associated semantic processing regions during code comprehension is particularly striking, because it suggests that the cognitive machinery used for understanding programming language is substantially the same machinery used for understanding natural language: a system organized around semantic interpretation, not logical inference. This does not mean that code comprehension is identical to natural language comprehension — the syntactic structure of programming languages is more rigid and the semantic conventions are more formally specified — but it suggests that the fundamental cognitive orientation is similar: both are semantic activities in which the reader constructs a model of meaning from a structured input, guided by prior knowledge of the conventions governing that input.

The working memory activation during code comprehension confirms the picture developed in the preceding sections. Understanding code requires maintaining an active representation of the current state of the computation — the scope, the values of variables in scope, the control-flow context — while parsing each new element of the code and updating the representation accordingly. This is exactly the kind of active maintenance-and-manipulation task for which the dorsolateral prefrontal cortex and associated working memory systems are engaged. The demands on these systems are higher when the code's structural organization does not support chunking — when scope boundaries must be tracked by counting indentation levels rather than reading explicit structural markers, when dependency relations must be inferred from non-local

syntactic cues rather than read from visible structure.

The brain imaging evidence thus provides neurological support for the Spherepop thesis: that scope visualization is not merely a pedagogical convenience but a cognitively significant feature of a notation system. Making scope explicit in the visual structure of the representation reduces the working memory load of scope tracking, freeing cognitive resources for other aspects of comprehension. The notation system is not an inert carrier of formal content; it is an active contributor to the cognitive process of understanding that content. A notation system designed without reference to the cognitive architecture of its users will impose unnecessary cognitive costs, even when it is formally adequate.

6.5 Spatial Reasoning and Symbolic Manipulation

The relationship between spatial reasoning and symbolic manipulation is one of the central puzzles in cognitive science, and it has direct implications for the design of notation systems. The traditional view, inherited from a broadly Kantian philosophical tradition, treats spatial intuition and logical-symbolic reasoning as distinct cognitive faculties: the former is a capacity for representing and reasoning about spatial relations among objects in the world, while the latter is a capacity for manipulating abstract symbols according to formal rules. On this view, the use of spatial representations in mathematics and programming — diagrams, graphs, geometric proofs — is a heuristic convenience that helps the spatial-intuitive faculty assist the logical-symbolic faculty, but the real cognitive work of

reasoning is done by the symbolic faculty operating on abstract representations.

This view has been substantially challenged by evidence from cognitive science and neuroscience. Spatial reasoning and symbolic reasoning share significant neural resources and interact in complex ways during mathematical and programming tasks. Mental rotation tasks — tasks requiring the spatial transformation of a represented object — activate many of the same brain regions as symbolic manipulation tasks, suggesting that spatial and symbolic representations are not as cleanly separated in the brain as the traditional view suggests. More directly relevant, the performance of mathematical tasks that would appear to be purely symbolic — mental arithmetic, algebraic manipulation — is systematically influenced by spatial variables: the spatial arrangement of numbers on a mental number line, the spatial organization of the work in a calculation, the directionality of written notation.

The implication for Spherepop is significant. If spatial and symbolic cognition are not distinct faculties but interacting systems that share resources and representations, then a notation system that makes the spatial structure of a computation explicit — that encodes scope as containment, evaluation order as nesting depth, dependency as spatial proximity — is not merely making a pedagogical accommodation to spatial intuition. It is recruiting spatial cognition into the computational task in a way that distributes the cognitive load across more of the available cognitive architecture, potentially reducing the overall burden while increasing the richness of the representation.

The nesting structure of Spherepop bubbles is not an arbitrary choice of visual metaphor. Containment is one of the most

primitive spatial relations available to human cognition — it is among the first spatial concepts that develop in infancy, before formal symbolic competence of any kind. A notation system grounded in containment therefore connects formal computational concepts to spatial intuitions that are developmentally primitive and neurologically robust. This is not a weakness of the notation — a concession to intuition at the expense of rigor — but a strength: it means that the notation can be understood by spatial cognition at the same time that it is being processed by symbolic cognition, providing a form of cognitive redundancy that supports accuracy and reduces error.

6.6 Semantic Grouping in Human Cognition

Human cognitive systems have a strong tendency to organize information into meaningful groups. This tendency — variously described as chunking, Gestalt organization, or categorical perception — operates across sensory modalities and cognitive domains, and it is one of the fundamental mechanisms by which the brain manages the information load of complex environments. In vision, the Gestalt principles of proximity, similarity, continuity, and closure describe the conditions under which elements of a visual scene are grouped into perceptual units. In language, the segmentation of continuous speech or text into words, phrases, and sentences reflects the operation of syntactic and semantic grouping mechanisms. In memory, the chunking of individual items into meaningful groups is the primary mechanism by which working memory capacity is effectively extended: items that can be grouped into a single meaningful chunk occupy a single slot in working memory

regardless of the complexity of their internal structure.

The relevance of semantic grouping to programming comprehension is direct. Code is not processed as a linear sequence of individual tokens but as a hierarchically organized structure of semantic units: expressions, statements, blocks, functions, modules. The grouping of code into these units is partly supported by syntactic structure — indentation, delimiters, keyword markers — and partly by the programmer's prior semantic knowledge of what kinds of units appear in code of the relevant type. When the syntactic marking of units is clear and consistent with the semantic grouping, comprehension is efficient: the reader's grouping perception is aligned with the code's actual semantic organization. When the syntactic marking is ambiguous, inconsistent, or requires active reconstruction from implicit conventions — as in the case of operator precedence in nested arithmetic expressions — the reader must perform additional cognitive work to recover the correct grouping, at the cost of working memory resources.

Spherepop's bubble notation is designed to align syntactic and semantic grouping perfectly. Each bubble is simultaneously a syntactic delimiter — it marks the boundary of a syntactic unit — and a semantic scope marker — it encodes the evaluative locality of the expression it contains. The Gestalt principle of closure operates naturally on closed bubble boundaries, causing the contents of each bubble to be perceived as a coherent unit. The principle of nesting — which is not a standard Gestalt principle but is closely related to the principle of enclosure — causes the hierarchical structure of the computation to be perceived as a hierarchical structure of visual units, with inner units clearly contained within outer units.

The perceptual organization of the diagram is aligned with the semantic organization of the computation, so that the grouping perception operates correctly without requiring conscious reconstruction.

This alignment between perceptual and semantic organization is not a trivial achievement. It is the result of a design principle that prioritizes cognitive legibility over notational conciseness: the willingness to use more visual space to make semantic grouping explicit, rather than compressing semantic grouping into syntactic conventions that must be actively decoded. The bet underlying Spherepop's design is that this trade-off is worthwhile: that the cognitive resources saved by eliminating the need for active grouping reconstruction outweigh the additional visual complexity of the explicit bubble notation.

6.7 Why Scope Visualization Matters

Scope — the region of a computation within which a variable, constraint, or evaluation rule is active — is one of the most fundamental structural concepts in programming language theory, and one of the most cognitively demanding for programmers to track correctly. Scope errors are among the most common sources of bugs in programs: uses of variables outside their scope, confusion between variables with the same name in different scopes, incorrect assumptions about which scope a variable reference resolves to. These errors are not primarily logical errors; they are errors in the programmer's internal model of the scope structure of the program. The programmer's mental representation of the scope structure is incorrect,

and the code fails because it is executed according to the actual scope rules rather than the programmer's mistaken model.

The cognitive difficulty of scope tracking arises directly from the invisibility of scope in standard linear notation. In a textual program, scope is typically marked by indentation, by delimiter characters such as braces or parentheses, or by keyword pairs such as `begin/end` or `do/end`. These markers are present in the text, but they do not make scope *visible* in any strong sense: a reader scanning the text must actively track the depth of delimiter nesting, maintain a model of what is currently in scope, and update that model as they encounter scope-opening and scope-closing markers. This is an active cognitive task that consumes working memory resources and is prone to error when the nesting depth is large or when multiple scopes overlap in complex ways.

Spherepop makes scope visible in a much stronger sense. The boundary of a bubble is a visible boundary of a scope region: everything inside the bubble is within the scope defined by the bubble, and everything outside is not. A reader does not need to track delimiter depth or maintain an internal scope model; they can read the scope structure directly from the visual containment structure of the diagram. A variable introduced within a bubble is visibly local to that bubble. A reference to an outer-scope variable visibly crosses bubble boundaries. The scope of the $\mathsf{bind}_{a<b}()$ operator is the interior of the bubble within which it is defined, and this scope is visible in the diagram without any need for active inference.

The practical consequence is not merely that scope errors are reduced — though that is a real benefit — but that the programmer's cognitive model of the program's structure be-

comes more accurate and more easily maintained, because it is grounded in a direct perceptual representation rather than in an abstract reconstruction. A programmer working with Spherepop notation does not merely know the scope structure abstractly; they see it. And seeing, in this context, is not a metaphor but a literal description of a cognitive process: the visual system is doing part of the cognitive work of scope tracking, offloading it from the working memory system and freeing those resources for other aspects of the comprehension task. Scope visualization matters because it changes the cognitive architecture of program comprehension, redistributing the cognitive load from the limited capacity of working memory to the much higher capacity of the visual system.

6.8 Nested Objects and Cognitive Compression

The ability to represent hierarchically nested structures — structures in which units contain other units, which contain still other units, to arbitrary depth — is one of the distinctive capacities of human symbolic cognition, and one that distinguishes it most sharply from the cognitive capacities of other species. Language is the paradigm case: the recursive syntax of human language permits the embedding of phrases within phrases, clauses within clauses, sentences within sentences, to any depth in principle, and this recursive embedding is the source of the extraordinary expressive power of language. Mathematics and programming language share this recursive structure: expressions may be nested within expressions, functions may call other functions, proofs may appeal to other proofs.

The cognitive management of nested structures is closely

tied to the chunking mechanisms described in the preceding section. A deeply nested structure can be made cognitively tractable by chunking: treating a complex nested substructure as a single unit with a known semantic role, and representing that unit as a single chunk in working memory. This is how experienced programmers read complex code: not by parsing every token sequentially but by recognizing familiar nested patterns — common idioms, design patterns, algorithms — and treating them as single meaningful units. The working memory cost of a complex nested structure is thereby reduced to the cost of a single chunk, provided the structure is familiar enough to be recognized as a chunk.

The critical insight from the chunking literature is that chunking requires the chunk's internal structure to be compressible: the chunk must have a semantic interpretation — a meaningful role in the computation — that can be activated as a unit and does not require the full internal structure to be maintained in working memory simultaneously. A deeply nested expression can be chunked only if its nested subexpressions have clear semantic roles that can be represented compactly. When they do not — when the nesting is purely syntactic, without clear semantic articulation — chunking fails and the full nested structure must be maintained in working memory, overwhelming its capacity.

Spherepop's bubble notation supports cognitive compression precisely by making the semantic articulation of nested structures visible. Each bubble is a natural chunking unit: it contains a semantically coherent subexpression, bounded by its bubble boundary, with a well-defined evaluation role within the larger computation. The history of the computation — the

sequence of bubble pops — is a sequence of chunk resolutions: each pop takes a chunked subexpression, evaluates it to a result, and substitutes the result for the chunk in the larger expression. The cognitive load at each step is the load of evaluating a single chunk, which is bounded by the complexity of the individual chunk rather than by the complexity of the full nested expression. This is cognitive compression in the strict sense: the representation reduces the working memory cost of maintaining the computation's state by chunking its hierarchical structure into manageable units with explicit semantic roles.

Chapter 7

The Biological Basis of Structural Thinking

Intelligence is characterized as predictive navigation through constraint space. The chapter develops a model of cognition as simulation and generative inference, examines mentalizing and recursive representation, and arrives at a conception of intelligence as structured compression that stands in sharp contrast to mere erasure.

7.1 Intelligence as Predictive Navigation

The dominant framework for understanding intelligence in contemporary cognitive neuroscience is predictive processing: the view that the brain is fundamentally a prediction machine, continuously generating predictions about its sensory inputs on the basis of internal models, comparing those predictions with the actual inputs, and updating the internal models when predictions fail. On this view, perception is not passive reception of sensory data but active inference: the brain proposes a model of the world and tests it against the incoming sensory stream,

refining the model in response to prediction errors. Cognition, memory, attention, and action are all understood as aspects of this predictive inference process, organized hierarchically across multiple levels of the cortical hierarchy.

The predictive processing framework has significant implications for the understanding of intelligence. Intelligence, on this view, is not primarily the ability to manipulate symbols according to formal rules, nor the ability to store and retrieve large amounts of information, nor even the ability to reason logically. It is the ability to build accurate predictive models of the world — models that generate correct predictions across a wide range of situations, that can be rapidly updated in response to new information, and that can be used to guide action effectively in environments whose future states must be anticipated in order to be navigated successfully. Intelligence is, at its core, the ability to navigate effectively through a structured possibility space by maintaining a model that predicts which possibilities are accessible from the current state.

This characterization of intelligence maps directly onto the Spherepop framework's understanding of computation. The admissibility manifold of a Spherepop computation — the space of all admissible evaluation sequences accessible from the current state — is the computational analogue of the possibility space through which an intelligent agent navigates. The Spherepop history operator $\Pi := e_n \circ \cdots \circ e_1$ is the computational analogue of the agent's trajectory through that space: the ordered sequence of actions by which it moves from initial state to goal state. And the admissibility conditions encoded in the bubble topology are the computational analogues of the environmental constraints that structure the agent's possibility

space, ruling out certain moves and permitting others.

Intelligence as predictive navigation is therefore not a metaphor for computation; it is the same structural operation described in two different domains. Both are processes of constrained trajectory selection through structured possibility spaces. Both require maintaining an accurate model of the current state, predicting which transitions are admissible from that state, selecting among admissible transitions according to some optimization criterion, and updating the model in response to the outcomes of past transitions. The Spherepop framework makes this structural identity explicit by using the same formal vocabulary — admissibility, history, constraint, trajectory — for both the computational and the cognitive case.

7.2 Modeling the World

The internal models that the predictive brain maintains are not simple lookup tables mapping inputs to outputs. They are generative models: structured representations of the causal structure of the environment, capable of generating predictions about what the world would look like under various hypothetical conditions, and capable of being run forward to simulate the consequences of hypothetical actions before those actions are taken. A generative model of a chess position, for instance, does not merely encode which positions are good; it encodes the causal structure of the game — the rules that determine which moves are legal from which positions, the strategic principles that determine which legal moves are likely to lead to advantageous positions — in a form that can be used to simulate the consequences of candidate moves before they are executed.

The structure of a generative model is directly relevant to the concerns of this monograph because it is a constraint structure: a representation of which world states are admissible from which other world states, under which conditions, and with what costs. The model constrains the space of predictions the brain can generate: predictions that violate the model's constraints are inadmissible and are not generated. And the model is updated — its constraints are revised — when predictions generated by the model fail to match incoming sensory data. This process of constraint revision is the fundamental cognitive operation of learning.

The parallel with Spherepop's admissibility conditions is exact. The bubble topology of a Spherepop expression is a constraint structure that determines which evaluation sequences are admissible: which pop events may occur at which stages, which bind constraints apply to which subexpressions, which refuse events are triggered by which conditions. The topology is updated — revised — when a pop event changes the constraint configuration: each pop removes the constraints associated with the popped bubble and may introduce new constraints in the containing bubble. Learning, in the Spherepop framework, is the process by which the constraint structure of the admissibility manifold is revised in response to the outcomes of past evaluations — exactly as the brain's generative model is revised in response to prediction errors.

7.3 Simulation and Generative Cognition

One of the central contributions of the predictive processing framework is the recognition that the brain does not merely re-

spond to the world; it simulates the world. Mental simulation — the internal generation of representations of hypothetical world states, using the brain's generative models — is a pervasive feature of human cognition that underlies planning, imagination, empathy, counterfactual reasoning, and language comprehension. When a person plans a route through an unfamiliar city, they run their internal model of the city's street network forward in imagination, simulating the consequences of different route choices before committing to one. When they understand a sentence describing an event, they partially simulate the described event, activating sensorimotor representations of the relevant objects and actions.

The cognitive function of mental simulation is the evaluation of possibilities before commitment. By simulating the consequences of a hypothetical action in the internal model, the agent can assess those consequences without incurring the real-world costs of actually performing the action and observing the outcome. Simulation is therefore a form of cheap exploration of the possibility space: it allows the agent to evaluate many more possibilities than could be explored through direct action, at a fraction of the cost. The quality of the agent's decisions is limited by the accuracy of the internal model used for simulation: a model that correctly predicts the consequences of actions permits accurate evaluation; a model that generates incorrect predictions leads to systematically poor decisions.

The connection to Spherepop is again direct. The evaluation of a Spherepop computation before commitment — the assessment of which evaluation trajectories are admissible, which have low action, which reach the desired terminal state — is a computational form of mental simulation. The admissibility

manifold plays the role of the internal world model: it encodes the causal structure of the computation, the constraints that determine which transitions are admissible, and the costs associated with each transition. Exploring the admissibility manifold before committing to a specific evaluation trajectory is the computational analogue of mentally simulating possible routes before beginning a journey. The principle of least structural commitment, developed in Chapter 12, is the computational analogue of the strategy of choosing the route whose simulated cost is lowest — not the route that is locally cheapest at each step, but the route whose total cost over the entire journey is minimized.

7.4 Mentalizing and Recursive Representation

Among the cognitive capacities that most sharply distinguish human intelligence from that of other animals is the capacity for mentalizing: the ability to represent the mental states of other agents — their beliefs, desires, intentions, and knowledge — and to use those representations to predict and interpret their behavior. Mentalizing, also called theory of mind, is a recursive representational capacity: to mentalize about another agent is to represent that agent as a system that itself represents the world, and to reason about the properties of that agent's representations. Mentalizing is therefore a second-order cognitive operation: cognition about cognition.

The recursive character of mentalizing is directly relevant to the concerns of this monograph because it is structurally identical to the recursive character of Spherepop's nested bubble notation. A bubble nested within a bubble is a scope within a

scope: a region of computation whose admissibility conditions are constrained both by the outer bubble's boundary conditions and by its own internal structure. The relationship between a nested bubble and its container is the same structural relationship as the relationship between an agent's representation and the meta-representation of that representation in the mind of a mentalizing observer: a structured embedding of one representational level within another, with constraints flowing from the outer level to the inner level and information flowing from the inner level to the outer.

This structural parallel has a deeper implication: that the cognitive capacity for recursive embedding — the ability to represent representations, to reason about reasoning, to model models — is not merely a specialized social cognitive capacity but a general feature of the computational architecture of the human mind that manifests across domains including language, mathematics, planning, and self-reflection. The nested bubble structure of Spherepop is not an idiosyncratic notational choice; it is a computational instantiation of a recursive representational architecture that is native to human cognition. This is one of the reasons why the Spherepop notation is immediately intuitive to many observers: it maps onto a cognitive structure that is already present, rather than imposing an alien structure that must be learned from scratch.

7.5 Prediction as Constraint Propagation

The predictive processing framework describes cognition as hierarchical Bayesian inference: each level of the cortical hierarchy generates predictions for the level below it, and the lower

level sends prediction errors — signals encoding the discrepancy between the prediction and the actual input — back to the higher level, which updates its model to reduce future prediction errors. The predictions flowing downward through the hierarchy are constraints on what the lower levels are expected to produce. The prediction errors flowing upward are signals that some constraint has been violated, triggering a revision of the constraint.

This hierarchical constraint propagation is structurally identical to the constraint propagation in Spherepop's bubble topology. The outer bubble constrains the inner bubble: the outer bubble's admissibility conditions restrict the set of admissible evaluation sequences within the inner bubble. When the inner bubble is popped, the result propagates upward to the outer bubble: the outer bubble's constraint configuration is updated to reflect the now-resolved inner expression. Prediction errors in the predictive brain are the cognitive analogues of refusal events in Spherepop: they are signals that the current constraint configuration is violated, triggering an update of the constraint structure — a revision of the generative model — that brings the constraint configuration back into consistency with the observed data.

Proposition 7.1. *In both the predictive brain and the Spherepop evaluator, the fundamental computational operation is constraint propagation: the downward propagation of constraints from higher levels to lower levels, and the upward propagation of constraint-violation signals from lower levels to higher levels. The dynamics of both systems are governed by the iterative resolution of constraint mismatches until a consistent constraint configuration is reached.*

This is not a casual analogy. It points to a genuine struc-

tural commonality between the computational architecture of biological intelligence and the computational architecture of Spherepop: both are constraint-propagation systems organized hierarchically, with top-down constraints and bottom-up error signals, converging on consistent constraint configurations through iterative refinement. The convergence condition — the condition at which the system reaches a stable state — is the condition of admissibility: the state in which all active constraints are satisfied simultaneously, all prediction errors are below threshold, all evaluation preconditions are met. In both cases, understanding is the condition of admissibility: the state in which the internal model is consistent with the available evidence.

7.6 Intelligence as Structured Compression

The predictive processing framework characterizes intelligence as the ability to build accurate generative models of the world. But there is a further structural property that distinguishes genuinely intelligent compression from mere memorization: the models built by intelligent systems are not merely accurate but structured. They capture the causal relations among world events, the constraints that make some events predictable from others, the invariants that hold across superficially different situations. A truly intelligent system does not merely remember that event A was followed by event B in the past; it understands why A was followed by B — what causal mechanism or structural regularity connects them — and can therefore predict that A will be followed by B in new situations where the relevant mechanism operates, even if those situations differ from the

past ones in other respects.

Structured compression differs from unstructured compression in precisely this way. Unstructured compression reduces a set of observations to a compact representation that encodes the observations faithfully but reveals no structure beyond the regularities needed for faithful reproduction. Structured compression reduces observations to a compact representation that encodes the causal and structural relations among them in a form that supports inference and prediction in novel situations. The first type of compression is what a lookup table achieves. The second type is what a causal model achieves. Intelligence is the capacity for structured compression.

The connection to the monograph's central distinction — between compression that preserves structure and compression that erases it — is immediate. The distinction is not between compression and non-compression; all cognition involves compression, because the world is vastly more complex than any internal representation. The distinction is between compression that preserves the structural relations — the causal connections, the constraint relations, the admissibility conditions — that support inference and generalization, and compression that preserves only the terminal outcomes while discarding the structural relations that generated them. Intelligence is the capacity for the former. The calculator, the black-box algorithm, and the opaque AI system are instances of the latter: they achieve terminal-state accuracy at the cost of structural transparency.

Spherepop's insistence on preserving evaluation history — on making the trajectory through the admissibility manifold visible rather than erasing it in favor of the terminal result —

is therefore not merely a preference for transparency. It is a design commitment to the form of compression that supports understanding, generalization, and genuine cognitive engagement with the computation being performed. The Spherepop representation of a computation is structurally richer than its terminal value in exactly the sense in which a causal model is structurally richer than a lookup table: it encodes the relations that make the terminal result intelligible, not merely the result itself.

Chapter 8

Compression Versus Erasure

The chapter draws a rigorous distinction between simplification and obliteration. Semantic memory and operational geometry are developed as complementary frameworks, and the notion of a cognitive map is recast as a constraint field—a topological structure over admissible inference paths.

8.1 The Difference Between Simplification and Obliteration

The distinction between simplification and obliteration is one of the most important conceptual distinctions in epistemology, and it is one that is systematically effaced by the vocabulary of compression, abstraction, and simplification that pervades the philosophy of science, mathematics, and computation. Both simplification and obliteration reduce the complexity of a representation. Both produce representations that are smaller, more concise, and more tractable than the original. Both are

described, in ordinary usage, as forms of compression or abstraction. But they differ in a respect that is decisive for the purposes of understanding: simplification reduces complexity while preserving the structural relations that make the original representation meaningful, while obliteration reduces complexity by discarding those structural relations.

The difference can be stated precisely in the language developed in Part III. A simplification of a representation X is a collapse $\mathsf{collapse}(q) : X \to X/{\sim_q}$ in which the equivalence relation $\sim_q$ is admissibility-preserving: the collapsed representation retains all the information necessary to determine which further operations on X are admissible. An obliteration of X is a map that discards admissibility information: the resulting representation does not support correct inference about the admissibility of further operations, because the structural relations that govern admissibility have been erased. A simplification is semantically lossless in the sense that matters for further reasoning. An obliteration is semantically lossy in ways that cannot be detected from the compressed representation alone.

The calculator provides the cleanest illustration. When a calculator evaluates $1 + 3 \times 2^2$ and returns 13, it performs a map from an expression with rich internal structure to a terminal value with no internal structure. This map discards all the admissibility information of the original expression: from the terminal value 13, it is impossible to determine what subexpressions were evaluated first, what constraint conditions were in force at each evaluation step, or what alternative evaluation orders would have been admissible. The map is obliteration, not simplification, because the information it discards is pre-

cisely the information required to understand why the result is 13 rather than some other value. The result is correct, but the representation of the process that produced it has been destroyed.

Contrast this with the Spherepop representation of the same computation: a sequence of bubble pops that records the evaluation order, the constraint configuration at each step, and the result of each pop. This representation is simpler than the original expression — shorter, more tractable — but it simplifies rather than obliterates: it compresses the description of the computation while preserving the structural relations that make the computation intelligible. The admissibility conditions are preserved: from the Spherepop history, it is possible to determine what was evaluated when, what was admissible at each step, and what the commitment cost of each transition was. The compression is lossy only in the sense of discarding what is genuinely redundant, not in the sense of discarding what is structurally essential.

The philosophical importance of this distinction extends well beyond the domain of computation. In scientific theory change, simplification is the replacement of a more complex theory by a simpler one that retains the structural relations needed to account for the phenomena the original theory explained. Obliteration is the replacement of a more complex theory by a simpler one that loses the structural relations, producing a representation that is formally adequate within a limited domain but fails to support the generalizations and predictions that the original theory supported. In education, simplification is the introduction of an accessible presentation of a concept that preserves its essential structure. Obliteration is the replacement

of the concept's essential structure by a formula that produces correct answers without conveying understanding. The distinction between simplification and obliteration is the distinction between teaching and training.

8.2 Semantic Memory and Operational Geometry

Semantic memory — the long-term memory system responsible for storing general knowledge about the world, as opposed to episodic memory's storage of specific personal experiences — is organized not as a list of facts but as a structured network of relations among concepts. Concepts in semantic memory are represented by their relations to other concepts: their categorical memberships, their functional roles, their typical properties, their typical contexts, their causal connections with other concepts. The meaning of a concept, in semantic memory, is constituted by its position in this relational network: to know the meaning of a word is to know how it relates to other words and concepts, which inferences it licenses, which contexts it is appropriate in, which other concepts it presupposes.

This relational structure of semantic memory is directly relevant to the Spherepop framework because it suggests that meaning, at the level of cognitive architecture, is inherently geometric: it is a matter of position in a structured relational space, not a matter of association with some intrinsic semantic content that exists independently of relations. A concept's meaning is its location in semantic space — the set of its distances and angular relations to all other concepts — and semantic inference is navigation through this space: moving from one concept to another along paths whose semantic relations support the

inference.

Operational geometry, as developed in this monograph, is the formal version of this picture applied to computation. The operational geometry of a Spherepop expression is the structure of its admissibility manifold: the set of admissible evaluation trajectories, their action costs, the constraint relations that govern them, and the terminal states they reach. The semantic content of the expression — what it means — is constituted by this operational geometry: by the structure of the computation it performs, the constraints it imposes, the evaluation history it traces. Two expressions have the same meaning, in the operationally geometric sense, if and only if they have the same admissibility manifold: the same space of admissible evaluation trajectories, with the same costs and the same terminal states.

This operational notion of meaning is closely related to the proof-theoretic notion of meaning developed in the philosophy of logic, on which the meaning of a logical connective is constituted by the rules governing its introduction and elimination in proofs, rather than by the truth-functional semantics of the connective. Both accounts locate meaning in operational structure rather than in correspondence with some external reality: meaning is what the expression does in the context of reasoning or computation, not what it refers to in some abstract semantic domain. The operational geometry of Spherepop provides a spatial and geometric realization of this operational notion of meaning, making explicit the structure that proof-theoretic accounts leave implicit.

The connection between semantic memory and operational geometry suggests a further possibility: that the semantic mem-

ory system in the brain may implement something like an operational geometry for the concepts it represents. Concepts in semantic memory are not merely lists of properties but structured representations of the causal and inferential relations among concepts — in effect, representations of the operational geometry of the concept in the space of cognitive operations. When the brain understands a new concept, it integrates it into this operational geometry: establishing its relations to existing concepts, determining which inferences it licenses, locating it within the constraint structure of the relevant knowledge domain. Understanding, on this view, is the construction of the operational geometry of a concept: the mapping of its position in the space of cognitive operations rather than merely the storage of a definition.

8.3 Cognitive Maps and Constraint Fields

The concept of a cognitive map, introduced by Tolman in his studies of rat navigation in the 1930s and subsequently extended to human cognition by a wide range of researchers, describes the internal spatial representation that organisms use to navigate their environments. A cognitive map is not a photographic record of sensory experience but a structured representation of spatial relations: distances, directions, landmarks, routes, and the topological connections among locations. It supports flexible navigation — the ability to take novel routes, to detour around obstacles, to find shortcuts — that cannot be explained by simple stimulus-response learning, because flexible navigation requires a representation of the spatial structure of the environment that can be used to plan routes that have

never previously been traversed.

The relevance of cognitive maps to the concerns of this monograph is that the concept of a map as a structured representation of relations — a representation that supports navigation and inference in a space of possibilities — generalizes far beyond literal spatial navigation. Cognitive maps in the broader sense are representations of the relational structure of any domain in which the agent must navigate: the conceptual space of a scientific domain, the social space of an organization, the problem space of a puzzle, the semantic space of a language. In each case, the cognitive map represents the admissibility relations in the domain — which transitions are possible from which states — and supports navigation through the domain by encoding the costs and constraints associated with different trajectories.

Definition 8.1. A constraint field $\mathcal{F}$ over a cognitive map $\mathcal{M}$ is a function that assigns to each point $x \in \mathcal{M}$ a constraint configuration $C(x)$: the set of admissibility conditions that govern transitions from x in the map. The constraint field encodes the topological structure of the admissibility manifold over the cognitive map: the way in which admissibility conditions vary across the space, creating regions of high and low accessibility, attractors and repellers, accessible and inaccessible regions.

A cognitive map equipped with a constraint field is a representation of the admissibility manifold of the domain being navigated: the space of all admissible trajectories through the domain, with the constraint field encoding which transitions are available from which locations. The constraint field is not uniform: some regions of the cognitive map are highly constrained, with few admissible transitions, while others are open,

with many available transitions. Navigation in the domain consists of selecting trajectories through the constraint field from the current location to the goal location, subject to the constraint conditions encoded by the field.

The constraint field formulation unifies several concepts that are typically treated separately. The scope structure of a Spherepop computation is a constraint field over the computational space: each bubble imposes a constraint on the evaluation transitions admissible within its interior. The admissibility manifold of a thermodynamic system is a constraint field over the state space: each constraint configuration determines which transitions are admissible from the current macrostate. The generative model of the predictive brain is a constraint field over the perceptual space: each prior expectation constrains which sensory inputs are predicted and therefore which prediction errors will be generated. In each case, the constraint field is the primary structured object: it determines the geometry of navigation through the space of possibilities, ruling out inadmissible trajectories and shaping the accessible routes through the domain.

The motif that recurs across all these domains — the motif established as one of the monograph's central structural claims in Chapter 12 — is expressed most completely here at the level of cognitive architecture: *intelligence navigates constraint fields*. The intelligent agent, the intelligent organism, and the intelligent computation all achieve their ends not by following predetermined routes through a structureless possibility space, but by reading the constraint field of their domain and selecting admissible trajectories that reach desired terminal states at acceptable cost. The structure of the constraint field is the

structure of the domain's intelligibility: the way in which some things are possible and others are not, some transitions cheap and others costly, some terminal states accessible and others foreclosed by the current constraint configuration. Understanding a domain is having an accurate internal representation of its constraint field. Intelligence is the capacity to navigate that field effectively.

Part III

Thermodynamics, Entropy, and Constraint

Chapter 9

The Frustration of Formula Manipulation

This chapter opens with a phenomenological account of the frustration produced by thermodynamics as typically taught: equations are known, processes are not. The black-box formalism of standard pedagogy is anatomized, and thermodynamics is reframed as a theory of structural transformation whose power lies precisely in the historical development of its concepts.

9.1 Knowing Equations Without Understanding Processes

There is a particular species of frustration, known to almost every student who has worked through an introductory thermodynamics course, that has no precise name but is immediately recognizable in retrospect. It is the experience of being able to manipulate a set of equations with reasonable fluency while remaining entirely uncertain what physical process those equations are actually describing. One can write $dU = \delta Q - \delta W$ and

proceed from there to calculate the work performed by an ideal gas undergoing a reversible isothermal expansion, arriving at the correct numerical answer, and yet feel, with an honesty difficult to suppress, that the calculation has passed through one's mind the way water passes through a channel: leaving no alteration behind. The channel is shaped by the equations; the water runs through without friction; nothing is learned about why the channel has that shape.

This is not a failure of intelligence. It is a structural consequence of how thermodynamics is typically presented. The student is handed a finished formal apparatus and instructed to operate it. The apparatus is correct, internally consistent, and extraordinarily powerful. But the process by which it was constructed has been systematically erased from the presentation, replaced by a series of definitions, axioms, and deductions that proceed in a logical order bearing almost no resemblance to the historical order of discovery or to the intuitive order in which the concepts actually become meaningful. The result is that the student acquires the compressed terminal notation without ever having inhabited the intermediate structure that gives that notation its depth.

Consider the simplest illustration available, one that will reappear at increasing levels of generality throughout this monograph. The expression $1 + 3 \times 2^2$ is a perfectly correct piece of mathematical notation. A calculator produces the value 13 essentially instantaneously. But the student who has memorized the operator precedence convention and applied it mechanically has done something importantly different from the student who actually sees why exponentiation must be resolved before multiplication, and why both must be resolved

before addition, not because of an arbitrary social convention, but because the nested dependency structure of the expression demands exactly that evaluation order. The first student knows the answer. The second student understands the shape of the computation. Thermodynamics, in the form in which it is typically taught, produces students of the first type.

9.2 The Problem of Black-Box Formalism

The term *black-box formalism* refers to any representational system in which the inputs and outputs are well-defined but the internal causal structure mediating them is constitutionally opaque. A calculator is a paradigm case. A thermodynamics textbook organized around definitions, derived relations, and worked examples can achieve the same effect at a higher level of abstraction. The student inputs a physical scenario and a set of constraints, applies the relevant equations, and extracts a numerical result. The machinery is reliable. The result is correct. The causal narrative of how the physical system actually moves through its state space over time — the trajectory, the sequence of constraints that become active and inactive, the way accessible volume in phase space expands or contracts — is nowhere visible in the formalism as presented. It has been collapsed into a set of symbols that perform the bookkeeping of the trajectory without displaying it.

The epistemological damage caused by black-box formalism is not primarily that it produces wrong answers. It rarely does. The damage is more subtle and more durable. It produces practitioners who are competent within the formalism but who cannot transfer their competence to problems that lie slightly

outside its canonical application domain, because what they have internalized is the surface syntax of the formalism rather than its underlying conceptual geometry. When the canonical templates fail to match the new problem, no internal resource is available for reconstruction from first principles, because those first principles were never part of what was transmitted.

There is a further pathology specific to thermodynamics that deserves attention. Thermodynamic formalism is unusually successful at permitting what might be called *structural bypasses*: formal derivations that reach correct conclusions without requiring the derivation to proceed through physically meaningful intermediate states. The classical derivation of the Carnot efficiency, for instance, can be reproduced as a sequence of algebraic manipulations without the student ever needing to visualize what actually happens to the gas during each stroke of the cycle, without feeling the constraint imposed by the second law as a geometric restriction on accessible trajectories in phase space rather than as a prohibition tacked onto the end of a calculation. The structural bypass is pedagogically efficient. It is cognitively impoverishing.

9.3 Thermodynamics as Structural Transformation

To say that thermodynamics is a theory of structural transformation is not to offer a metaphor. It is to make a precise claim about what thermodynamic laws actually govern. The first law is a statement about the conservation of a certain quantity — internal energy — under the transformations that constitute thermodynamic processes. The second law is a statement about

the asymmetry of those transformations: that not all transformations are equally available at all times, that the space of futures accessible from a given state is constrained in a particular way, and that this constraint is not incidental to the theory but is its deepest content. What the second law really says, at the level of conceptual structure rather than formal statement, is that *the set of admissible futures shrinks in a characteristic direction under spontaneous evolution*.

This is a claim about topology. Not about differential equations, not about macroscopic variables like temperature and pressure considered as functions of time, but about the shape of the space of possibilities and how that shape changes as physical processes unfold. The entropy of a system in a given macrostate is, on the statistical interpretation developed by Boltzmann and subsequently clarified by Gibbs and Shannon, a measure of the volume of that space: specifically, the logarithm of the number of microstates compatible with the macroscopic constraints currently imposed on the system. A high-entropy macrostate is one accessible from an enormous number of microscopic configurations. A low-entropy macrostate is one accessible from relatively few.

Thermodynamic evolution, then, is the process by which a system moves through the space of admissible macrostates in response to the removal or relaxation of constraints. When a partition separating two gases at different temperatures is removed, the constraint is lifted, and the system evolves toward states accessible from a much larger volume of microstate space. The new equilibrium macrostate is not merely more probable in some abstract statistical sense. It is the attractor of a dynamical process, the terminus of a trajectory through admissibility

space, and the entropy increase that accompanies it is precisely the expansion of that space as the lifted constraint ceases to exclude the newly accessible configurations.

Reframing thermodynamics in this way is not merely aesthetic. It reveals that the theory is, at its structural core, a theory of constrained reachability: of which futures are accessible from which presents, and how the topology of that accessibility changes as constraints are applied, modified, or removed. This is exactly the same conceptual structure that Spherepop makes visible in computation, the same structure that appears in the Spherepop history operator $\Pi := e_n \circ \cdots \circ e_1 : X_0 \to X_n$, where the composition records not merely the terminal state but the entire ordered sequence of transitions through admissible intermediate states, each transition constrained by the admissibility conditions in force at the moment of evaluation. The connection between computation and thermodynamics is not analogical. It is structural.

Chapter 10

The Historical Development of Energy

History is argued to be constitutive of concept formation in physics. The chapter traces the emergence of energy conservation from mechanics through heat, culminating in the statistical interpretation of entropy and the realization that constraint is the core object of thermodynamic theory.

10.1 Why History Matters for Understanding Concepts

The history of a scientific concept is not merely biographical decoration appended to a presentationally convenient formal definition. For a certain class of concepts — and energy is the paradigm case — the history is constitutive of the concept's meaning, because the concept does not exist prior to the network of relations, tensions, and partial disambiguations that its history records. A student who encounters the definition of energy as "the capacity to do work" and who then proceeds to apply that definition mechanically to a series of exercises has

not thereby acquired the concept of energy. They have acquired a local handle on a compressed summary of a concept whose depth is inaccessible from that summary alone. The depth becomes accessible only through the sequence of conceptual crises and partial resolutions that the historical development of the concept traverses.

This is not a philosophical eccentricity. It is a claim about how conceptual compression works. When a concept that originally had genuine referential depth — that was attached to specific phenomena, specific confusions, specific partial models that were discarded or revised — gets compressed into a clean formal definition, the compression is lossy in a precise sense. It preserves the extensional content: the conditions under which the term correctly applies. But it erases the intensional content: the web of relations, contrasts, and structural pressures that determine why the concept has the shape it has rather than some other shape. The formal definition is a quotient of the concept's history under an equivalence relation that identifies all paths to the same result. The quotient is efficient. The kernel of the quotient map — the structure being collapsed — contains the understanding.

Spherepop's $\mathsf{collapse}(q)$ operator makes this process explicit: it takes a trajectory through admissible intermediate states and maps it to an equivalence class, with the quotient map preserving only what the equivalence relation $\sim_q$ declares observable. The question of what gets lost in a given collapse is precisely the question of what conceptual depth is sacrificed when a historical scientific concept is compressed into its terminal formal definition.

10.2 From Mechanics to Heat

The concept of energy as it appears in modern thermodynamics is the product of a convergence between two initially independent research traditions: the mathematical mechanics descending from Newton and Leibniz, and the practical engineering tradition concerned with the relationship between heat and mechanical work that emerged primarily from the study of steam engines in the late eighteenth and early nineteenth centuries. These two traditions used different languages, addressed different phenomena, and were organized around different theoretical purposes. Their convergence was not smooth. It involved genuine conceptual surprises, false starts, and retrospective reinterpretation of earlier results whose significance was not recognized at the time of their production.

Within the Newtonian tradition, the conserved quantity was momentum, $\mathbf{p} = m\mathbf{v}$, and the study of its transfer in collisions and gravitational interactions was the central mathematical project. Leibniz's alternative proposal — that the conserved quantity was *vis viva*, proportional to mv^2 — was the object of a prolonged and somewhat acrimonious dispute. What the dispute revealed, in retrospect, was that both sides were correct about something real: linear momentum and kinetic energy are both conserved in elastic collisions, and they transform differently under changes of reference frame, encoding different aspects of the system's dynamical structure. The dispute was not resolvable by direct confrontation because the two quantities being championed did not compete directly. They inhabited different levels of the theory's structure.

This failure of simple competition between mv and mv^2 is itself a structural lesson. It illustrates the general principle that

when two formal quantities both appear to track something real, the correct response is usually not to select one and discard the other but to recognize that they are projections of a richer structure onto different observable directions. The richer structure, in this case, is the full phase-space portrait of the mechanical system, of which momentum and kinetic energy are both partial encodings. A notation system that forces selection of one projection and discards the others is performing exactly the kind of lossy collapse that leaves the student with a correct answer and no understanding of why that answer is correct.

10.3 The Discovery of Conservation Laws

The emergence of energy conservation as a general law — applicable not only to purely mechanical systems but to systems involving heat, chemical transformation, electromagnetic effects, and eventually nuclear processes — was the result of a series of conceptual advances that were, in several important cases, arrived at simultaneously and independently by investigators working in different countries, different disciplines, and different theoretical frameworks. Mayer, Joule, Helmholtz, and Colding each contributed to the recognition that a single conserved quantity mediates transformations among apparently different physical phenomena. The fact of simultaneous independent discovery is not merely historically interesting. It is theoretically significant, because it suggests that the concept of energy conservation was not an arbitrary theoretical choice but was, in some sense, already latent in the structure of the phenomena waiting to be articulated.

What the early investigators were struggling to express was

the intuition that physical processes are transformations of something, not creations or destructions of it, and that the something being transformed is a quantity whose total amount is preserved across the transformation. The difficulty was that the quantity appeared in radically different forms in different domains: kinetic energy in mechanics, heat in thermal systems, electrical energy in electromagnetic systems, chemical potential in reactions. Each form had its own local quantitative measure, and the theoretical work required to establish that these different measures were all aspects of a single conserved quantity was considerable.

Joule's careful experimental work on the mechanical equivalent of heat established the quantitative exchange rate between mechanical work and thermal energy, providing the empirical foundation for the equivalence. But the conceptual work of recognizing that this equivalence implied a universal conservation law required the additional insight that the different apparent forms of energy were not fundamentally different substances but different modes of the same underlying process. Heat, it became clear, is not a substance (caloric) but a form of mechanical motion at the microscopic level. This recognition did not merely add a new entry to a list of energy forms. It changed the ontological status of the conservation law from an empirical generalization about macroscopic quantities to a structural consequence of the microscopic constitution of matter.

10.4 Entropy and Statistical Interpretation

The statistical interpretation of entropy, developed principally by Boltzmann in the 1870s and 1880s and subsequently refined

by Gibbs, represents perhaps the deepest conceptual advance in the history of thermodynamics, and also one of the most pedagogically mishandled. The formula $S = k_B \ln W$, where W is the number of microstates compatible with the observed macrostate, is routinely presented to students as though it were a definition — a decision to measure entropy in units of k_B times the logarithm of a count. But this is not what the formula is. It is a theorem: a demonstration that the macroscopic quantity S defined by Clausius in terms of heat flows at constant temperature is identical, up to a constant factor, to the logarithm of the microstate count. The demonstration requires substantial theoretical work, and the result is not trivial. It says that the macroscopic arrow of time — the direction in which entropy spontaneously increases — is a consequence of the fact that high-W macrostates are overwhelmingly more probable than low-W macrostates, so that a system undergoing spontaneous evolution will almost certainly move from macrostates of lower probability to macrostates of higher probability, simply as a consequence of the combinatorics of microstate counting.

The conceptual gain from the statistical interpretation is enormous, because it relocates entropy from the status of a phenomenological quantity — defined by what it does in macroscopic processes — to the status of a structural quantity: a measure of the volume of a region in the space of microscopic states. Entropy becomes, on this interpretation, a geometric object: the logarithm of an accessibility volume. And thermodynamic evolution becomes the movement of a system through regions of that space in response to constraints that either confine it to small regions or permit it to explore large ones.

Definition 10.1. Let Γ denote the phase space of a system with

fixed macroscopic parameters. For a macrostate M, let $\Omega(M) \subset \Gamma$ denote the region of phase space consistent with M. The Boltzmann entropy of M is

$$S(M) = k_B \ln|\Omega(M)|,$$

where $|\cdot|$ denotes the phase-space measure (Liouville measure) of the region. Thermodynamic evolution is the process by which the effective constraint on the system's microstate evolves, moving the system's representative point through regions of Γ of progressively larger measure.

This definition is entirely standard, but its geometric restatement has consequences that standard presentations rarely draw out. The increase of entropy during spontaneous irreversible processes is not merely a statistical tendency. It is the expansion of the accessible volume in phase space as constraints are relaxed. When a gas expands into a vacuum, the volume of phase space available to the system's microstate increases because the spatial extent of the available positions has increased. The entropy increase records this expansion. The process is irreversible not because nature has a preference for disorder but because the expanded region of phase space is enormously larger than the contracted one, and spontaneous fluctuations back to the contracted region are correspondingly rare.

10.5 Constraint as the Core of Physics

If entropy measures accessible volume, and thermodynamic evolution consists in the expansion of that volume as constraints are relaxed, then the primary objects of thermodynamic theory are not states or variables but *constraints* themselves. This is

a significant shift of emphasis. Standard presentations foreground the state variables — pressure, temperature, volume, internal energy, entropy — and treat the constraints (fixed walls, insulating boundaries, permeable membranes) as boundary conditions imposed on the equations governing those variables. But a more penetrating analysis reveals that the constraints are prior. The state variables are meaningful only relative to a specification of what constraints are in force, and the laws of thermodynamics are most naturally read as statements about how accessible volume in constraint space evolves under the operations of constraining and de-constraining.

Consider a simple example. A gas confined to one half of a container by a removable partition occupies a macrostate characterized by its temperature, pressure, volume, and internal energy. When the partition is removed — when the constraint is lifted — the gas evolves to a new equilibrium that occupies the full container. The new macrostate has higher entropy. But what has physically changed? The gas itself has not changed in any intrinsic way. What has changed is the set of admissible microstates: before removal of the partition, only microstates with all molecules in the left half were admissible; after removal, microstates with molecules anywhere in the full container are admissible. The entropy increase is entirely a consequence of this expansion of the admissibility set.

This framing — constraint as the primary object, state as the secondary object defined by a constraint set — is exactly the framing that Spherepop adopts for computation. In Spherepop, the current state of a computation is not merely a value but a pair consisting of a value and the set of admissible operations that may be performed on it: the evaluation region, the scope,

the admissibility conditions encoded in the local bubble structure. The bind operator $\mathsf{bind}_{a<b}(X)$ is not merely a syntactic annotation; it is a constraint that restricts the admissibility domain for subsequent operations. The refuse operator $\mathsf{refuse}(e)$ is not a failure mode; it is the formal record of a constraint that prevents a particular reduction from being admitted. Computation, on the Spherepop view, is constraint navigation, exactly as thermodynamic evolution is constraint navigation. The structural analogy is not imposed from outside. It is internal to the conceptual architecture of both theories.

Chapter 11

Admissibility, Entropy, and Thermodynamic Histories

Entropy is reinterpreted as a measure of accessibility volume in the space of admissible states. Collapse and renormalization are treated as geometric operations. The chapter introduces thermodynamic histories as the natural objects of a Spherepop-style action formalism.

11.1 Admissible States and Future Possibility

The concept of admissibility, as it appears in Spherepop, is a generalization of a notion that appears in several branches of physics under different names. In classical mechanics, the analogue is the constraint surface: the submanifold of phase space on which a constrained system's trajectory is required to lie, defined by the holonomic or non-holonomic constraints imposed by the system's physical configuration. In thermodynamics, the analogue is the equilibrium manifold: the subspace of state space corresponding to macrostates that satisfy the equilibrium conditions imposed by the constraints currently in force. In

statistical mechanics, the analogue is the energy shell: the hypersurface in phase space on which the total energy takes a prescribed value, to which the microstate is restricted when the system is isolated.

In each case, the admissibility set is not a fixed background feature of the theory but a dynamic object that changes as constraints are applied or removed. When a constraint is applied — a wall inserted, a chemical bond formed, a temperature fixed — the admissibility set contracts. When a constraint is released — a wall removed, a bond broken, a thermal reservoir disconnected — the admissibility set expands. The evolution of the admissibility set under constraint operations is the primary dynamical object of thermodynamic theory, more fundamental than the evolution of individual state variables.

Definition 11.1. Let $\mathcal{C}$ denote a set of constraints on a physical system, and let Γ denote the system's phase space. The admissibility set $\mathsf{Adm}(\mathcal{C}) \subseteq \Gamma$ is the subset of phase space consistent with all constraints in $\mathcal{C}$. The future possibility space of a state $x \in \mathsf{Adm}(\mathcal{C})$ under spontaneous evolution is the set of states reachable from x by dynamical trajectories that remain within $\mathsf{Adm}(\mathcal{C})$.

The entropy of a macrostate, on the statistical interpretation, is therefore directly a measure of the future possibility space available to the system. High entropy means large future possibility space. Low entropy means small future possibility space. The second law — that entropy increases spontaneously in isolated systems — says that systems naturally evolve toward states with larger future possibility spaces, because such states are overwhelmingly more probable in the space of all accessible trajectories. This is not a deep metaphysical fact about the

direction of time. It is a combinatorial fact about the sizes of regions in phase space.

What is profound is what this implies about the relationship between constraint and freedom. Applying a constraint to a system reduces its entropy (or, more precisely, prevents the entropy from growing along the trajectories that the constraint rules out). A perfectly constrained system — one for which every degree of freedom has a fixed prescribed value — has zero entropy, because its admissibility set contains a single point: its future possibility space is empty. A completely unconstrained system has maximum entropy: every microstate is admissible, and the future possibility space is the entire phase space. Real physical systems occupy the vast middle ground, and the history of their evolution is the history of how their admissibility sets expand and contract as constraints are applied by external interventions and released by spontaneous thermalization.

11.2 Entropy as Accessibility Volume

The geometric reframing of entropy as accessibility volume has several consequences that bear directly on the Spherepop framework. The first is that entropy becomes a property not of individual states but of admissibility sets, and therefore of constraint configurations. This is already implicit in Boltzmann's formula, but standard presentations obscure it by writing S as a function of macrostate variables, as though entropy were a property of the macrostate considered in isolation rather than a property of the macrostate-under-constraint. The correct reading of $S(T, V, N)$ is: the entropy of a system whose macroscopic degrees of freedom take values T, V, N, under the implicit con-

straint that these values are held fixed. Different constraints, differently specified, yield different entropy values for the same underlying microscopic configuration.

Proposition 11.2. *Let $C_1 \subset C_2$ be constraint sets with C_1 more restrictive than C_2, so that $\mathsf{Adm}(C_1) \subseteq \mathsf{Adm}(C_2)$. Then the entropy under C_1 is no greater than the entropy under C_2:*

$$S(C_1) = k_B \ln|\mathsf{Adm}(C_1)| \;\leq\; k_B \ln|\mathsf{Adm}(C_2)| = S(C_2).$$

The proof is immediate from the set inclusion. But the conceptual content is nontrivial: entropy is monotonically decreasing in constraint strength. Adding constraints always reduces or preserves entropy. Removing constraints always increases or preserves entropy. The second law of thermodynamics, in this framing, says that isolated systems evolve by spontaneously relaxing internal constraints — constraints that were not externally imposed but arose from the system's initial conditions — and that this relaxation produces entropy increase because it expands the accessibility volume.

This framing also clarifies the relationship between equilibrium and entropy. Equilibrium is the condition in which the internal constraints have all relaxed and the system's admissibility set is as large as the externally imposed constraints permit. At equilibrium, entropy is maximized subject to the external constraints. This is the content of the maximum entropy principle, which now reads: a system at equilibrium occupies the largest accessible volume in phase space that is compatible with its external constraints. The approach to equilibrium is the process of exploring that volume, which proceeds spontaneously because the vast majority of trajectories from any

low-entropy state lead rapidly to higher-entropy states simply as a consequence of the phase-space geometry.

The accessibility volume interpretation also connects entropy to the theory of information in a way that is more than analogical. Shannon's entropy formula,

$$H = -\sum_i p_i \log p_i,$$

measures the uncertainty in a probability distribution over possible states. When the distribution is uniform over an admissibility set of cardinality W and zero outside it, Shannon entropy reduces to $\log W$, which is proportional to Boltzmann entropy. The connection is not a coincidence. Both measures quantify the same structural fact: how much of the possibility space is accessible from the current epistemic or physical position. Entropy is simultaneously a thermodynamic quantity, a measure of accessibility volume, and a measure of remaining uncertainty about the system's microstate. These are not three separate interpretations; they are three descriptions of the same geometric object.

11.3 Collapse and Renormalization

The operation of $\mathsf{collapse}(q)$ in the Spherepop formalism — the mapping $X \to X/\sim_q$ that identifies trajectories under an admissibility-preserving equivalence relation — has a direct and important analogue in thermodynamics: the operation of coarse-graining, which plays a central role in the renormalization group and in the general theory of how macroscopic descriptions emerge from microscopic ones.

Coarse-graining is the operation of partitioning a phase space into cells of some resolution and identifying all microstates within a cell as observationally equivalent. The result is a reduced description of the system in terms of which cell it occupies rather than which precise microstate. The coarse-grained entropy of a macrostate is the logarithm of the number of cells intersecting the macrostate's accessibility volume, rather than the logarithm of the number of individual microstates. When the cells are large, coarse-grained entropy is lower than fine-grained entropy, because many microstates are identified as equivalent and the effective count is reduced.

Definition 11.3. Let $q : \Gamma \to \Gamma/\sim_q$ be a coarse-graining map that partitions phase space into equivalence classes $[x]_q$. The coarse-grained entropy is

$$S_q(M) = k_B \ln|\{[x]_q \mid x \in \Omega(M)\}| = S(M) - k_B \ln|\ker q|,$$

where $\ker q$ denotes the typical fiber of the map, and the subtraction records the information lost by identifying microstates within each equivalence class.

The formula $S_q = S - k_B \ln|\ker q|$ is the thermodynamic version of the Spherepop collapse renormalization:

$$\mathsf{collapse}(q) : S \mapsto S - \log|\ker q|.$$

In both cases, the collapse reduces apparent complexity by the logarithm of the kernel size. In both cases, the reduction is structured rather than arbitrary: the equivalence relation $\sim_q$ is required to be admissibility-preserving, meaning that the quotient map respects the observable consequences of the

trajectories being identified. And in both cases, the crucial conceptual point is that the collapse is not an annihilation of information but a structured forgetting: the kernel of the quotient map records precisely what is being identified, so the collapse is invertible in principle (the kernel can be recovered from the map) even when it is irreversible in practice (once the identification has been made, the distinguished microstates are no longer tracked).

The renormalization group in quantum field theory and statistical mechanics works by exactly this mechanism. A series of coarse-graining operations, each identifying short-distance degrees of freedom as equivalent, produces a sequence of effective theories at progressively longer length scales. Each step in the sequence is a collapse in the Spherepop sense: a quotient of a fine-grained theory by an equivalence relation that identifies the short-distance fluctuations being integrated out. The effective coupling constants at each scale are the coarse-grained counterparts of the fine-grained Lagrangian parameters. The renormalization group flow is the trajectory traced by the effective theory through the space of possible theories as the coarse-graining scale increases.

What the renormalization group reveals, and what the Spherepop formalism makes explicit, is that the relationship between theories at different scales is not one of simple approximation but of structured projection: each coarser theory is a quotient of the finer theory by a precisely characterized equivalence relation, and the information lost in the projection is exactly the logarithm of the kernel of that relation. This is compression in the strict sense: representation of a complex object by a simpler one, with the compression ratio precisely

measured by the entropy of the equivalence relation.

11.4 Thermodynamic Histories

The natural objects of thermodynamic theory, once the constraint-navigation framework is adopted, are not states but histories. A thermodynamic history is an ordered sequence of constraint configurations through which a system passes, together with the accessible volumes associated with each configuration and the transitions between them. The entropy of the system at each moment is a property of the constraint configuration at that moment, not an intrinsic property of the microstate. The entropy change between two moments is the change in accessibility volume, which depends on what constraint operations intervened between them.

Definition 11.4. A thermodynamic history of length n is a sequence

$$\mathcal{H} = ((C_0, M_0),\ (C_1, M_1),\ \ldots,\ (C_n, M_n)),$$

where each C_k is a constraint configuration, each M_k is the macrostate at step k, and consecutive pairs $(C_{k-1}, M_{k-1}) \to (C_k, M_k)$ are connected by either a constraint operation (application or removal of a constraint) or a spontaneous equilibration within $\mathsf{Adm}(C_{k-1})$.

The entropy profile of the history is the sequence $S(C_0), S(C_1), \ldots, S(C_n)$. The second law constrains this profile: for steps that are spontaneous equilibrations (no external constraint operation), entropy must be non-decreasing. For steps that are constraint operations, entropy may increase or decrease, depending on

whether the operation relaxes or applies a constraint.

The importance of histories over states is directly parallel to the importance of histories in the Spherepop formalism. In Spherepop, a computation is not a function from input to output but a trajectory through the space of admissible evaluation states, each pop event changing the constraint configuration and thereby the set of further admissible evaluations. The result of a computation is the terminal state of the history, but the history itself carries information — about how the result was reached, what alternatives were foreclosed at each step, what the computational cost of each transition was — that the terminal state alone does not contain. The terminal value of $1 + 3 \times 2^2$ is 13; the Spherepop history records that this value was reached by first collapsing the exponentiation bubble, then the multiplication bubble, then the addition bubble, in that specific order, and that each collapse was admissible precisely because no outer bubble's constraint blocked it.

The analogy between thermodynamic and computational histories is not merely structural. It points toward a deeper claim: that what we mean by *understanding* a physical or computational process is precisely having access to the history, not merely to the terminal state. A student who has been given only the final answer 13 knows nothing about the computational process. A student who has watched the sequence of bubble pops understands why the answer is 13 and why no other answer was possible given the expression's constraint structure. Understanding, in both thermodynamics and computation, is access to the ordered sequence of admissibility transitions that constitute the process.

Chapter 12

The Spherepop Action Formalism

Computational trajectories are developed as the analogues of physical paths in configuration space, and the variational principle of stationary action is extended to a principle of stationary semantic commitment. The path-integral picture of quantum mechanics is reread as a structural analogy for interpretation: stable meanings are regions of constructive interference over admissible semantic histories, while incoherent interpretations destructively interfere and cancel. Action is then interpreted as computational cost, commitment as reduction of remaining freedom, and Hamiltonians of admissibility as the generators of constrained evolution. The chapter closes with treatments of information as constraint and semantic entropy. The motif reality selects histories, not merely states *is established here as one of the monograph's central structural claims.*

12.1 Histories as Computational Trajectories

In classical mechanics, the primary objects of the Lagrangian and Hamiltonian formalisms are not individual states but trajec-

tories through configuration space or phase space. The action $S[\gamma] = \int_\gamma L\,dt$ assigns a real number to each trajectory γ, and the physically realized trajectory is the one that extremizes this functional — the trajectory that makes the action stationary under variations. This principle of stationary action is not merely a computational device. It is a structural claim about the relationship between the space of possible trajectories and the space of physically admissible ones: the admissible trajectories are precisely those at which the action functional is stationary, and the value of the action along such a trajectory encodes the system's dynamical cost of traversing it.

The Spherepop action formalism extends this variational structure to the domain of computational histories. A computational trajectory in the Spherepop framework is an ordered sequence of evaluation events

$$\gamma = (e_1, e_2, \ldots, e_n), \quad e_k \in \{\mathsf{pop}(\cdot)\,,\ \mathsf{refuse}(\cdot)\,,\ \mathsf{collapse}(\cdot)\,,\ \mathsf{bind}_{\cdot}(\cdot)\},$$

where each event is admissible relative to the constraint configuration in force at the moment of its application. The space of all such sequences, over all admissible orderings and all admissible selections of events, constitutes the space of possible computational trajectories. The action functional assigns to each trajectory a cost that measures, in a unified way, both the thermodynamic cost of performing the evaluation events and the structural cost of the commitment each event entails.

Just as in classical mechanics, the set of physically (or computationally) realized trajectories is a proper subset of the set of all possible trajectories: not every sequence of evaluation events is admissible, not every admissible sequence has the same cost, and the structural properties of the system — its

bubble topology, its constraint configuration, its admissibility conditions — determine which trajectories are accessible from a given starting state. The history operator $\Pi := e_n \circ \cdots \circ e_1$ is the Spherepop analogue of the path integral measure: a weighted sum over all admissible trajectories that connect the initial state X_0 to the final state X_n, where the weight of each trajectory is determined by its action.

12.2 Variational Computation and Stationary Histories

The principle of stationary action in classical mechanics does not merely assert that physical systems follow the path of least action; it asserts something more subtle and more general: that the physically realized trajectory is distinguished from all imaginable trajectories by a structural property — stationarity of the action functional under infinitesimal variations — that is global over the trajectory rather than local at any single point. No information about a single instant of the trajectory is sufficient to determine whether that trajectory is physically admissible. The trajectory must be known as a whole before its admissibility can be assessed. This is why the principle of stationary action is not a local law of motion, even though it produces local equations of motion through the Euler-Lagrange procedure: the locality of the resulting equations is a derived fact, not a primitive one. The primitive fact is global, and it concerns the structure of the trajectory considered as a unified object.

This global character of the action principle is what motivates the present section's intervention. Spherepop computation has been treated, in most of what precedes, as a locally

governed process: each pop event is admissible or inadmissible relative to the constraint configuration in force at the moment of its application. That local characterization is correct as far as it goes, but it leaves undescribed a structural feature of computation that the action formalism is precisely suited to capture: the way in which the globally optimal trajectory through admissibility space may differ from the trajectory selected by purely local greedy evaluation. A greedy evaluator always takes the locally cheapest pop event at each step. A variational evaluator selects the trajectory whose total action $S[\gamma] = \sum_t \mathcal{L}_t$ is stationary over the space of all admissible trajectories connecting the initial to the terminal state.

Definition 12.1. Let $\mathcal{A}(X_0, X_n)$ denote the space of all admissible computational trajectories connecting the initial state X_0 to the terminal state X_n. The *variational problem of computation* is to find

$$\gamma^* = \underset{\gamma \in \mathcal{A}(X_0, X_n)}{\arg\,\text{stationary}}\, S[\gamma],$$

the trajectory at which the action functional $S[\gamma]$ is stationary under admissible variations: perturbations $\gamma \to \gamma + \delta\gamma$ that preserve the boundary states X_0 and X_n and remain within the admissibility space $\mathcal{A}$.

The stationary trajectory γ^* is the computationally optimal history: the sequence of evaluation events that achieves the required transformation from X_0 to X_n at minimum total cost in future accessibility and structural commitment. It need not be unique — there may be multiple distinct trajectories at which the action is stationary, just as there may be multiple locally extremal paths in a mechanical system with symmetry — but each stationary trajectory satisfies the Euler-Lagrange equations for

the computational Lagrangian, which in the discrete setting become a condition on the local structure of each evaluation step: how the commitment cost ΔC_t and the accessibility change $\Delta\Omega_t$ must be balanced at each step in order for no admissible variation to reduce the total action.

The admissibility manifold $\mathcal{A}(X)$ of a semantic or computational object X is the space of all admissible trajectories originating from X: the set of all ordered sequences of evaluation events, beginning from the state X, that satisfy the constraint conditions encoded in the bubble topology $\mathcal{B}$ active at X. Histories are then paths $\gamma : \{0, 1, \ldots, n\} \to \mathcal{A}(X)$, and the action functional $S[\gamma]$ is a function on this path space. The variational problem is to identify which paths are stationary points of S.

12.2.1 The Principle of Least Structural Commitment

The admissibility-manifold formulation permits a restatement of the action principle in language closer to the monograph's philosophical concerns. Among all the trajectories through $\mathcal{A}(X_0, X_n)$ that connect the same initial and terminal states, the stationary trajectories are those that balance, across all steps simultaneously, the rate at which they convert future accessibility into present commitment. A trajectory that commits too aggressively at early steps forecloses evaluation alternatives that might have permitted a cheaper path to the terminal state. A trajectory that defers commitment too conservatively accumulates deferred evaluation costs that eventually must be paid. The stationary trajectory threads between these failure modes.

This is the *principle of least structural commitment*: admissible computation proceeds not by greedily resolving every available expression at the earliest opportunity, nor by deferring every

resolution until it can be avoided no longer, but by distributing commitments across the trajectory in the way that minimizes total structural cost. The principle is not a prescription for a specific evaluation order in any given computation. It is a characterization of the globally optimal evaluation strategy, which depends on the full structure of the admissibility manifold and cannot be determined by local inspection alone.

The principle of least structural commitment is the computational analogue of the principle of least action in mechanics. Both assert that the realized trajectory through a structured possibility space is distinguished by a global optimality condition. Both imply that understanding a process requires access to the trajectory, not merely to the terminal state. And both resist the reduction of the selection principle to a local rule: the global optimality condition cannot be restated as a condition on each individual step without reference to the trajectory's overall structure. This is precisely why opaque computational systems — systems that expose only terminal values — are inadequate for genuine understanding. They suppress the one object, the trajectory, that carries the information the variational principle makes essential.

12.2.2 Semantic Configuration Spaces

The notion of configuration space in mechanics — the abstract space whose points represent the instantaneous configurations of a physical system, and whose paths represent the system's trajectories through time — has a direct semantic counterpart in the Spherepop framework. The semantic configuration space of a computation is the space of all possible semantic states the computation can occupy during its evaluation: all possible

combinations of partially reduced expressions, active constraint configurations, and accumulated evaluation histories.

Formally, if X is a Spherepop expression with bubble topology $\mathcal{B}$, its semantic configuration space is the pair $(\mathcal{A}(X), \mathcal{B})$: the admissibility manifold together with the constraint structure that defines which paths through the manifold are admissible. A computation is a path through this space. The initial point of the path is the unreduced expression X_0 with its full bubble topology. The terminal point is the fully reduced expression X_n with all bubbles popped and all constraints resolved. The intermediate points are the partially reduced states through which the computation passes, each characterized by the subset of bubbles that have been popped, the current constraint configuration, and the accumulated commitment cost.

The semantic configuration space has a natural topology induced by the admissibility relation: two states are close if they differ by a small number of evaluation steps, and the path metric in this topology is the action $S[\gamma]$ of the shortest admissible path connecting the two states. This topology encodes the structure of the evaluation process: states that are reachable from each other by low-cost trajectories are close; states separated by high-cost mandatory transitions are distant. The semantic configuration space is therefore not merely a formal construction but a geometric object whose topology encodes the computational cost structure of the expression being evaluated.

The calculator that evaluates $1 + 3 \times 2^2$ provides no information about the semantic configuration space of the expression: it maps the initial state to the terminal state with no intermediate structure visible. Spherepop, by contrast, exposes the trajectory through the semantic configuration space: the exponentiation

bubble pops first, moving the computation from the initial state to a state in which 2^2 has been resolved; the multiplication bubble pops next, moving from that state to one in which 3×4 has been resolved; the addition bubble pops last, moving to the terminal state 13. Each pop is a step in the semantic configuration space, and the sequence of steps is the computation's trajectory through that space. The trajectory carries information about the evaluation order, the constraint structure at each step, and the commitment cost of each transition — information that the terminal value alone cannot recover.

12.2.3 Computational Geodesics

In a Riemannian manifold, geodesics are the curves of shortest length between two points: the paths that minimize the total distance traversed in the manifold's geometry. In a pseudo-Riemannian manifold such as spacetime, timelike geodesics are the paths that maximize proper time, which in the context of general relativity means the paths followed by freely falling observers. In both cases, geodesics are defined by a variational condition — extremality of a length or action functional — and in both cases, the geodesic equation, which is a local differential equation governing the path, is derived from the global variational condition by the calculus of variations.

The stationary trajectories of the Spherepop action functional are the computational analogues of geodesics in semantic configuration space. They are the paths of least structural commitment through the admissibility manifold: the evaluation orders that distribute commitment cost most efficiently across the steps of the computation, avoiding unnecessary foreclosure of future possibilities at early stages while not deferring manda-

tory commitments indefinitely. As in the geometric case, these stationary trajectories satisfy a local equation — the computational Euler-Lagrange equation — that is derived from the global variational principle and that determines, at each step, how the commitment cost and accessibility change must be balanced.

The notion of a computational geodesic makes precise the intuition that there is, for any given computation, an evaluation order that is in some sense natural: not arbitrary, not determined by convention (as operator precedence rules are determined), but geometrically determined by the structure of the expression's admissibility manifold. The conventional evaluation order for $1 + 3 \times 2^2$ — exponentiation first, then multiplication, then addition — is the geodesic of the semantic configuration space defined by the expression's nested bubble structure, because that order minimizes the total commitment cost: each bubble is popped at the earliest moment at which it is admissible to do so, which is also the moment at which popping it incurs the lowest cost in foreclosed future possibilities. The precedence rule, on this view, is not an arbitrary convention but a compressed encoding of the geodesic structure of arithmetic's semantic configuration space — and this is why the convention is non-arbitrary even though its presentation is typically unmotivated.

12.2.4 Histories as Variational Objects and the Path-Integral Analogy

The variational formulation of the Spherepop action suggests a further analogy, more speculative but philosophically significant, with the Feynman path-integral formulation of quantum

mechanics. In Feynman's formulation, the probability amplitude for a quantum system to transition from an initial state to a final state is computed not by identifying the single classically optimal trajectory but by summing contributions from all possible trajectories, weighted by a phase factor $e^{iS[\gamma]/\hbar}$ that depends on the classical action $S[\gamma]$ of the trajectory. The classical limit $\hbar \to 0$ recovers the principle of stationary action, because in this limit the contributions of non-stationary trajectories destructively interfere and cancel, leaving only the contribution of the stationary trajectory — the classical path — which interferes constructively with its near neighbours.

The conceptual content of the path integral formulation is that the physically realized outcome is not the result of a single trajectory being selected by some prior mechanism and then executed; it is the result of a superposition over all possible trajectories, with the classical behaviour emerging from the constructive interference of near-stationary trajectories and the suppression of non-classical behaviour by the destructive interference of trajectories with rapidly varying phases. The trajectory is not chosen; it is selected by the geometry of the interference pattern in path space.

The semantic analogy — and it must be insisted upon that this is an analogy, not a claim about quantum mechanisms in cognition — is the following. When a cognitive system or an interpretive framework encounters an ambiguous expression, it does not, typically, execute a deterministic selection procedure that identifies the uniquely correct interpretation and discards all others. It evaluates many possible interpretive trajectories simultaneously, and the stable interpretation that emerges is the one that is reinforced by constructive interference across

the space of admissible interpretations — the interpretation to which many nearby trajectories through semantic configuration space lead, and from which small perturbations of the initial expression or context do not depart. Unstable or incoherent interpretations correspond to trajectories that destructively interfere: small variations in context or expression lead to radically different interpretations, producing instability in the interpretation across the space of relevant variations.

Definition 12.2. A semantic interpretation m of an expression X is *stable under admissible variation* if there exists a neighbourhood U of X in semantic configuration space such that all trajectories γ with $\gamma(0) \in U$ and $\gamma(n) = X_n$ converge to the same terminal interpretation m. Stable interpretations are the semantic analogues of classically realized trajectories: they are the interpretations for which the space of nearby admissible histories is coherent, all leading to the same result.

The relevance of this analogy to the monograph's concerns is immediate. Opaque computational systems — calculators, black-box algorithms, language models that provide terminal outputs without visible reasoning trajectories — suppress all information about the interference structure in semantic configuration space. They provide the terminal interpretation without any indication of whether that interpretation is stable under admissible variation, whether nearby expressions would lead to radically different outputs, or whether the space of near-stationary semantic trajectories is coherent or fragmented. Spherepop's insistence on preserving the evaluation history is, in this light, an insistence on preserving the information needed to assess semantic stability: whether the terminal state $X_n = 13$ is robust to small variations in the expression, or whether it

is an accident of the specific evaluation path taken through a fragmented admissibility space.

This is the deepest form of the claim that underlies the entire Part III: *reality selects histories, not merely states*. In classical mechanics, the principle of stationary action selects the physical trajectory from the space of all possible ones; the terminal state is meaningful only as the endpoint of that trajectory. In quantum mechanics, the path integral selects stable outcomes as those that survive constructive interference across the space of all trajectories; the terminal probability amplitude is meaningful only as the integrated contribution of the full path-space geometry. In thermodynamics, the second law selects the direction of spontaneous evolution as the direction of increasing accessibility volume; the terminal equilibrium state is meaningful only as the terminus of the history of constraint relaxation that produced it. In Spherepop computation, the variational principle of least structural commitment selects the optimal evaluation trajectory from the admissibility manifold; the terminal value is meaningful only as the product of the specific history of admissibility transitions that generated it.

The structural pattern is the same in every case. What varies is the domain — physical, quantum mechanical, thermodynamic, computational — and the specific form taken by the action functional and the admissibility conditions. What is invariant is the logical structure: the trajectory is the primary object; the terminal state is a derived object; and understanding consists in access to the former, not merely the latter. A philosophy of computation that privileges terminal states over trajectories — that treats the output of a computation as its primary content and the computational history as an imple-

mentation detail — inverts this order of priority and thereby forfeits the possibility of genuine understanding of the processes it describes.

12.3 Action and Computational Cost

The Spherepop action functional is defined as

$$S[\gamma] = \sum_{t=1}^{n} \mathcal{L}_t,$$

where $\mathcal{L}_t$ is the Lagrangian at step t:

$$\mathcal{L}_t = \Delta\Omega_t + \lambda \Delta C_t.$$

Here $\Delta\Omega_t$ denotes the change in admissible possibility volume at step t, measured by the change in the logarithm of the number of admissible future evaluation sequences; ΔC_t denotes the change in commitment complexity, measured by the increase in the number of constraints actively binding future evaluations; and $\lambda > 0$ is a coupling constant that sets the relative weight of thermodynamic accessibility cost against structural commitment cost.

The Lagrangian $\mathcal{L}_t$ is therefore a sum of two conceptually distinct contributions. The first, $\Delta\Omega_t$, is a thermodynamic term: it records how much the evaluation step at time t contracts or expands the space of admissible future computations. A pop event that collapses a deeply nested bubble will typically reduce Ω significantly, because it forecloses many alternative evaluation paths that would have been available if the bubble had been left intact. A bind event that applies a new constraint

reduces Ω by exactly the volume of the phase space region ruled out by the new constraint. A refuse event preserves Ω locally at the cost of deferring the evaluation.

The second contribution, $\lambda \Delta C_t$, is a structural term: it records the increase in the system's commitment complexity — the number of active binding constraints — as a consequence of the evaluation step. Commitment complexity increases with each bind operation and decreases with each pop that resolves a bound constraint. It is a measure of the structural rigidity of the ongoing computation: how many constraints are currently in force, how much of the future evaluation space has been locked down by prior commitments, how much freedom of maneuver remains.

The sum $\mathcal{L}_t = \Delta \Omega_t + \lambda \Delta C_t$ therefore measures the total cost of a single evaluation step: the thermodynamic cost of reducing future accessibility plus the structural cost of increasing current commitment. A minimal-action trajectory is one that achieves its computational goal — reaches the terminal state — while incurring the smallest total cost summed over all steps. This is a precise computational analogue of the principle of least action in physics.

12.4 Commitment and Remaining Freedom

The conjugate momentum to the possibility volume Ω is defined, in direct analogy with Hamiltonian mechanics, as the partial derivative of the Lagrangian with respect to the rate of change of Ω:

$$\pi_t = \frac{\partial \mathcal{L}_t}{\partial \dot{\Omega}_t}.$$

For the Lagrangian $\mathcal{L}_t = \Delta\Omega_t + \lambda\Delta C_t$, this yields $\pi_t = 1$ as a formal statement in the discrete case, but the more illuminating expression emerges when the Lagrangian is taken to depend on the rate of change of accessibility volume: then π_t is the sensitivity of the Lagrangian to marginal changes in how quickly the accessibility volume is being contracted. This is the *commitment momentum*: the rate at which the evaluation process is converting future freedom into present determination.

The Hamiltonian is then:

$$H_t = \pi_t \Omega_t - \mathcal{L}_t.$$

On the physical interpretation, H_t represents the *remaining freedom* of the computation at step t: the product of the current accessibility volume Ω_t with the commitment momentum π_t, less the current Lagrangian cost. This is precisely the Legendre transform of the Lagrangian with respect to Ω, and it generates the Hamiltonian equations of motion for the constrained evolution:

$$\dot{\Omega}_t = \frac{\partial H_t}{\partial \pi_t}, \qquad \dot{\pi}_t = -\frac{\partial H_t}{\partial \Omega_t}.$$

The first equation says that the rate of change of accessibility volume is determined by the commitment momentum: systems with high commitment momentum evolve rapidly toward low-Ω states, collapsing their future possibility spaces quickly. The second equation says that the rate of change of commitment momentum is determined by the sensitivity of the Hamiltonian to changes in accessibility volume: when remaining freedom decreases steeply with volume, commitment momentum grows, reflecting the accelerating pressure toward resolution as the evaluation approaches its terminal state.

The philosophical content of the Hamiltonian formulation is significant. It identifies *remaining freedom* as the primary dynamical quantity of a computation: not the current state, not the terminal result, but the structured capacity for future determination that the current constraint configuration preserves. A computation with high remaining freedom is one that has not yet committed to a specific evaluation path: many alternatives are still accessible, many pops are still admissible, many bind operations could still be applied or not applied. A computation with low remaining freedom has locked itself into a nearly determined trajectory: the terminal result is nearly fixed by the accumulated commitments, and only a small number of admissible evaluation sequences remain.

This framing makes explicit what black-box computational systems conceal. When a calculator evaluates $1 + 3 \times 2^2 = 13$ instantaneously, it destroys all information about the remaining-freedom profile of the computation: how much future freedom was available after the exponentiation was resolved, how much remained after the multiplication, what the commitment cost of each step was. The result 13 is the terminal state of a trajectory through admissibility space whose remaining-freedom profile has been completely erased. Spherepop preserves this profile by making each evaluation step visible as a pop event with associated changes in Ω and C.

12.5 Hamiltonians of Admissibility

The Hamiltonian of admissibility H_t is not a single operator but a family of operators parameterized by the constraint configuration in force at step t. Different constraint configurations

— different bubble topologies, different sets of active bind constraints — yield different Hamiltonians, and therefore different equations of motion for the remaining-freedom dynamics. This is the Spherepop analogue of the observation in constrained Hamiltonian mechanics that constraint surfaces generate modified equations of motion through the Dirac bracket construction.

Definition 12.3. Let $\mathcal{B}_t$ denote the bubble topology in force at step t of a computation: the set of active bubbles, their nesting relations, and their associated admissibility conditions. The Hamiltonian of admissibility is the functional

$$H[\mathcal{B}_t](\Omega, \pi) = \pi\Omega - \mathcal{L}[\mathcal{B}_t](\Omega, \Delta C),$$

where $\mathcal{L}[\mathcal{B}_t]$ is the Lagrangian computed with respect to the constraint structure encoded in $\mathcal{B}_t$.

The evolution of $\mathcal{B}_t$ as computation proceeds is discrete: each pop event removes a bubble from the topology, each bind event adds a constraint to an existing bubble's boundary conditions, and each refuse event leaves the topology unchanged while recording the refusal as an event in the evaluation history. The sequence of bubble topologies $\mathcal{B}_0, \mathcal{B}_1, \ldots, \mathcal{B}_n$ constitutes the structural skeleton of the computational trajectory, and the sequence of Hamiltonians $H[\mathcal{B}_0], H[\mathcal{B}_1], \ldots, H[\mathcal{B}_n]$ is the sequence of generators of the remaining-freedom dynamics at each step.

The admissibility condition on each pop event is expressible as a constraint on the Hamiltonian: the event $\mathsf{pop}(b)$ at step t is admissible if and only if the bubble b is an innermost bubble in $\mathcal{B}_{t-1}$ — no other admissible bubble is nested strictly

inside b — and the constraint conditions on b's boundary are satisfied by the current state. This is the Spherepop analogue of the admissibility condition for Dirac first-class constraints in constrained Hamiltonian mechanics: a constraint is first-class if and only if it commutes with all other constraints under the Poisson bracket, meaning that its application is consistent with maintaining the other constraints. A bubble is poppable if and only if its internal evaluation is consistent with the constraint conditions imposed by all its containing bubbles.

12.6 Shannon and Statistical Structure

The connection between the Spherepop action formalism and information theory emerges most clearly when the accessibility volume Ω is interpreted not as a volume in a continuous phase space but as a count of admissible evaluation sequences in the discrete setting of combinatorial computation. In this setting, Ω_t is the number of distinct admissible computational trajectories that could complete the evaluation from state t onward, given the constraint configuration $\mathcal{B}_t$.

Shannon's entropy of the uniform distribution over these trajectories is then $H_t = \log \Omega_t$, which is precisely the Boltzmann entropy of the accessible region in the space of computational histories. The Spherepop action $S[\gamma] = \sum_t \mathcal{L}_t$ is therefore a generalization of the Shannon entropy summed over the trajectory: it measures not merely the uncertainty at each step but the total cost, in units of information and structural commitment, of the specific trajectory γ through the space of all admissible trajectories.

This connection makes precise the sense in which a compu-

tation has an informational cost. Each pop event that commits to a specific reduction is, from the perspective of the space of admissible trajectories, a selection among many possible evaluations: it reduces the remaining uncertainty about how the computation will proceed. The informational cost of the pop event is the reduction in $\log \Omega$ it produces: the number of bits by which the set of remaining possible completions is contracted. A pop event that resolves an exponentiation within a deeply nested context may contract the space of completions by many bits, because the result of the exponentiation flows through many subsequent operations. A pop event that resolves an isolated leaf expression may contract the space by only a few bits, because few subsequent operations depend on it.

12.7 Compression and Meaning

The information-theoretic interpretation of the Spherepop action raises a natural question: what is the relationship between compression of a computation and preservation of its meaning? This question has a precise answer within the Spherepop framework. Two computational histories γ_1 and γ_2 are semantically equivalent — they preserve the same meaning — if and only if they agree on all observable quantities: the terminal result and the admissibility structure of the evaluation (which constraints were in force at each step, which evaluations were admitted or refused). Histories that differ only in unobservable details — in the precise enumeration of microscopically distinct but observationally equivalent intermediate states — may be identified under the collapse operation $\mathsf{collapse}(q)$ without semantic loss.

Compression of a computational history is therefore the operation of identifying all semantically equivalent histories under an appropriate equivalence relation $\sim_q$. The compressed description $\gamma/\sim_q$ retains the observationally essential features of the trajectory — its terminal result, its admissibility profile, its commitment cost — while eliding the observationally redundant details. The compression ratio is $\log|\ker q|$, exactly as in the thermodynamic case. And the condition for the compression to be semantically valid is the factorization condition from the original Spherepop formulation:

$$f(x) = f(x') \wedge q(x) = q(x') \;\Rightarrow\; \exists! \bar{f} \text{ such that } f = \bar{f} \circ \mathsf{collapse}(q)\,,$$

which says that any observable function f that is invariant under $\sim_q$ descends uniquely through the collapse map. This is a categorical statement: it asserts the existence of a unique factorization through the coequalizer of the equivalence relation, which is the standard universal property of quotient objects in a category with quotients.

The distinction between compression that preserves meaning and compression that destroys it is therefore not a matter of degree but of structure. Compression preserves meaning if and only if the equivalence relation $\sim_q$ is admissibility-preserving: histories that are identified under $\sim_q$ must agree on all observationally meaningful features. Compression destroys meaning when the equivalence relation is applied without regard to admissibility, when histories with different observable features are identified because they share a terminal result, when the computational provenance is erased because the terminal value is all that is considered relevant. The calculator that returns 13 is performing compression of the second, destructive

kind. Spherepop insists on compression of the first, meaning-preserving kind.

12.8 Information as Constraint

The framing of information as constraint — rather than as content, as message, or as reduction of uncertainty — is the natural completion of the thermodynamic interpretation developed in this part of the monograph. On the constraint interpretation, information is not a substance or a quantity possessed by a message in isolation but a relational property: the information in a message, relative to a receiver's prior state, is precisely the set of constraints that message imposes on the receiver's future states. A message that rules out many possible futures carries more information than one that rules out few, because it more severely restricts the receiver's admissibility set.

This is Shannon's insight, stated in the language of the present framework. Shannon's entropy $H = -\sum_i p_i \log p_i$ measures the expected reduction in admissibility set that a message from a source with probability distribution p will impose on the receiver. When all messages are equally probable, the entropy is maximized: each message is maximally informative because each rules out the largest number of alternatives. When one message is certain, the entropy is zero: the message carries no information because it rules out nothing — the receiver already knows it is coming.

Definition 12.4. The information content of an event e relative to a constraint configuration C is

$$I(e \mid C) = \log \frac{|\mathsf{Adm}(C)|}{|\mathsf{Adm}(C \cup \{e\})|} = S(C) - S(C \cup \{e\}),$$

the reduction in entropy — the logarithm of the ratio of accessibility volumes — produced by conditioning on the occurrence of e.

On this definition, every pop event in a Spherepop computation carries a definite information content relative to the constraint configuration in which it occurs: the reduction in the accessibility volume of the remaining computation produced by the commitment the pop event entails. The total information content of a computational history $\gamma = (e_1, \ldots, e_n)$ is the sum of the information contents of its constituent events:

$$I(\gamma) = \sum_{t=1}^{n} I(e_t \mid C_{t-1}),$$

which is exactly the action $S[\gamma]$ when the Lagrangian is taken to be the pure information cost with $\lambda = 0$. The action functional thus generalizes the information content of a trajectory to include both the information-theoretic cost (the reduction in accessibility volume) and the structural cost (the increase in commitment complexity), weighted by the parameter λ.

12.9 Semantic Entropy

The final concept of this chapter, and of Part III as a whole, is *semantic entropy*: the analogue, in the domain of meaning and interpretation, of thermodynamic entropy in the domain of physical states. Semantic entropy measures the volume of the accessibility space in the domain of meaningful interpretations rather than physical microstates, and it obeys the same qualitative laws as thermodynamic entropy: it increases when constraints on interpretation are relaxed, decreases when con-

straints are applied, and is maximized at semantic equilibrium — the condition in which all meaningful interpretations compatible with the available evidence are equally accessible.

The concept of semantic entropy makes precise a set of intuitions that are otherwise difficult to articulate. When a text or expression is highly ambiguous — when many different interpretations are compatible with its surface form — its semantic entropy is high. When a text is highly specific — when its interpretation is almost uniquely determined by its context, prior knowledge, and semantic constraints — its semantic entropy is low. The process of interpretation is the process of reducing semantic entropy by applying constraints derived from context, prior knowledge, pragmatic implicature, and compositional structure. Understanding is the terminal state of this process: the condition in which the semantic admissibility set has been reduced to a small enough volume that a specific interpretation can be adopted with confidence.

Definition 12.5. Let $\mathcal{I}(e)$ denote the space of possible interpretations of an expression e, and let C denote the contextual constraints applicable to e (background knowledge, semantic conventions, pragmatic expectations). The semantic entropy of e relative to C is

$$S_{\text{sem}}(e \mid C) = \log|\{i \in \mathcal{I}(e) : i \text{ is admissible under } C\}|.$$

Semantic entropy is directly relevant to the critique of black-box computational systems developed in Chapter 9. When a calculator returns 13, the semantic entropy of the output, relative to the constraints provided by the input expression $1+3\times2^2$, is zero: there is exactly one admissible interpretation of the

numeral 13 in this context. But the semantic entropy of the *process* has been artificially collapsed: the calculator's output provides no information about which of the many possible computational trajectories was actually taken, what admissibility conditions were in force at each step, or what the commitment cost of the computation was. The output is semantically determinate but computationally opaque.

Spherepop's contribution to semantic entropy is structural: by preserving the evaluation history as a sequence of bubble pops with associated admissibility conditions and commitment costs, it maintains the computational information that the terminal value discards. The semantic entropy of a Spherepop computation history is higher than the semantic entropy of its terminal result, in the same way that a thermodynamic history carries more information than the macrostate at its terminal point. But this higher entropy is not noise or ambiguity — it is preserved structure, the record of how the result was reached, which is precisely the information required for understanding rather than merely for verification. Thermodynamics, information theory, and the Spherepop formalism converge here on a single principle: meaning is not in the terminal state alone but in the trajectory that reaches it.

Part IV

Emergence, Complexity, and Self-Organization

Chapter 13

The Emergent Universe

Complexity and self-organization are examined through the lens of nested systems: evolutionary constraint formation, emergence as structured collapse, and the cosmos as a constraint-navigation process. The chapter develops the idea that agency is a thermodynamic gradient phenomenon.

13.1 Complexity and Self-Organization

The study of complexity and self-organization is, at its most precise, the study of how constraint structures generate ordered patterns from disordered initial conditions. A self-organizing system is not a system that violates thermodynamics by producing order from nothing. It is a system that exploits the local admissibility conditions imposed by its physical substrate to channel the dissipation of free energy into the formation and maintenance of structured, far-from-equilibrium patterns. The order that emerges is not imposed from outside; it arises from the interaction of the system's components under the constraint of their local admissibility conditions, and it persists as long as

the free-energy gradient driving its formation continues to be supplied.

Complexity, in the technical sense relevant here, is not synonymous with disorder or unpredictability. A complex system is one that exhibits rich, structured behavior at multiple scales simultaneously: local interactions that generate global patterns, global constraints that shape local dynamics, and feedback loops that couple the two levels in ways that produce behaviors not predictable from either level alone. The Spherepop framework is naturally suited to the description of complexity in this sense, because the nested bubble structure of an admissibility manifold is precisely the kind of multi-scale hierarchical architecture that complex systems exhibit: local constraint configurations within individual bubbles, global constraint structures at the level of the full bubble topology, and coupling between levels through the propagation of pop results from inner to outer bubbles.

13.2 Nested Systems Within Systems

The recursive nesting of systems within systems is the organizational principle common to the most complex structures in the known universe: cells within organisms, organisms within ecosystems, ecosystems within planetary systems, social institutions within civilizations, propositions within proofs, subroutines within programs, bubbles within bubbles. In each case, the nested structure is not merely a convenient description imposed by an observer; it reflects a genuine architectural feature of the system: the fact that certain collections of components are more tightly coupled to each other than to components out-

side the collection, and that the interface between the collection and its environment is constrained in specific ways that make the collection's behavior at least partially autonomous from the external context.

The Spherepop framework treats nested systems as its primary formal object. Each bubble is a nested system: a collection of internal components (its contents) enclosed within a boundary (its membrane) that interfaces with the external context (the containing bubble or the outermost evaluation environment) through a constrained channel (the pop result). The nesting relation among bubbles directly encodes the nesting relation among the corresponding subsystems. The admissibility conditions at each bubble's boundary encode the interface constraints between the subsystem and its containing system. The pop operation encodes the upward propagation of computed results from inner subsystems to outer ones. The entire multi-scale architecture of a complex system is therefore directly representable in the Spherepop formalism, without loss of the hierarchical structure that makes the system's complexity organized rather than chaotic.

13.3 Evolutionary Constraint Formation

The evolution of complex biological systems can be understood, through the Spherepop lens, as a process of progressive constraint formation: the historical accumulation of admissibility conditions that successively narrow the space of viable developmental and behavioral trajectories while simultaneously enabling the reliable production of increasingly complex organized structures. Each evolutionary innovation — a new

biochemical pathway, a new structural protein, a new regulatory mechanism — is the addition of a new constraint to the organism's admissibility manifold: a new condition that the organism's developmental and physiological processes must satisfy in order to be viable.

This framing appears to be in tension with the standard account of evolution as the expansion of phenotypic possibility through the accumulation of heritable variation. But the tension is superficial. The paradox of evolutionary complexity is that the accumulation of constraints enables rather than restricts the production of complex structures. A protein whose folding is highly constrained — whose amino acid sequence is tightly coupled to a specific three-dimensional structure — can perform enzymatic functions that an unconstrained polymer cannot, precisely because the constraint enables a specificity and reliability of function that unconstrained polymers lack. Each new constraint added by evolutionary selection is a new organizational resource: a new admissibility condition that channels the dynamics of the system into reliable production of functional structures.

The evolutionary accumulation of constraints is therefore an instance of the more general phenomenon of path-sensitive historical modification of admissibility structure. The organism at any evolutionary stage carries in its genome, epigenome, and developmental programs the accumulated record of the constraints that proved viable in its lineage's history. This record is not merely a description of what happened; it is a constraint on what can happen next. The organism's future evolutionary trajectory is shaped by its accumulated admissibility structure: certain mutations are viable because they are compatible

with the existing constraint configuration, and others are lethal because they violate constraints that the organism's function requires.

13.4 Emergence as Structured Collapse

Emergence — the appearance of properties at a higher level of organization that are not present in the components at lower levels — is one of the most contested concepts in the philosophy of science. The Spherepop framework offers a deflationary but precise account: emergent properties are properties of the equivalence classes produced by the collapse operations that aggregate lower-level components into higher-level entities. They are properties of the quotient structure, not of the individual elements being quotiented.

The temperature of a gas is the paradigm case. Temperature is a property of the gas as a whole — a property of the macrostate — that is not a property of any individual molecule. It is a property of the equivalence class of microstates that are identified by the thermodynamic coarse-graining operation: the collapse $\mathsf{collapse}(q)$ that identifies all microstates consistent with the observed macroscopic parameters. Temperature is the quotient structure's property, not the components' property; it emerges from the collapse, not from the components themselves.

This account of emergence as structured collapse dissolves the apparent mystery of emergence without reducing it to nothing. The emergent property is real: temperature genuinely describes something about the gas that no description of individual molecules captures. But it is not mysterious: it is

precisely the property of the quotient structure produced by a specific, well-defined collapse operation. The emergence is structured rather than arbitrary because the collapse operation is admissibility-preserving: it identifies only microstates that are equivalent under the relevant thermodynamic observables, and it produces a quotient structure that inherits the admissibility conditions from the level below. Emergent properties are the invariants that survive the collapse.

13.5 The Romance of Reality

There is a view, associated with a tradition of scientific naturalism running from Democritus through Lucretius to the modern neo-Darwinian synthesis, that the discovery of the physical basis of complex phenomena is a disenchantment: that explaining biological organization in terms of chemistry, or mental processes in terms of neuroscience, or social structures in terms of individual psychology, is to reveal that the apparent richness and depth of these phenomena is illusory — that they are "nothing but" their material substrate, that the higher-level descriptions are convenient fictions rather than genuine features of reality.

The Spherepop framework rejects this view, not on mystical grounds but on formal ones. The claim that emergent properties are quotient structures rather than independent substances is not the claim that they are "nothing but" the components they emerge from. A quotient structure is not identical to the structure it is quotiented from; it has different properties, different admissibility conditions, and different relationships to other structures at the same level of organization. The temper-

ature of a gas is not the kinetic energies of its molecules; it is the equivalence class of all microstates with the same mean kinetic energy, and it has properties — it determines heat flow direction, it equals other gases' temperatures at equilibrium, it obeys the zeroth law — that no individual molecule's kinetic energy has.

The reality of emergent structures is the reality of quotient structures: not less real than their components, but differently real — real in a different sense, at a different level of description, with different admissibility conditions governing what is possible at that level. The structured richness of biological, cognitive, social, and cultural phenomena is not an illusion overlaid on a featureless physical substrate. It is the genuine formal structure of the quotient spaces that physical processes produce through their history of constraint formation and structured collapse. Reality is structurally rich at every level of the collapse hierarchy, and the romance of reality is the romance of that structural richness — the inexhaustible complexity of quotient structures that emerge from the history of admissibility-guided transformation.

13.6 Cosmic Complexity and Information

The universe as a whole can be understood, through the framework developed in Part III, as a vast history of constraint formation and release: a trajectory through the space of admissible physical configurations, beginning from an extremely low-entropy, highly constrained initial state and evolving through the successive formation and elaboration of structure — gravitational, nuclear, chemical, biological, cognitive — as the con-

straints of the initial state propagate forward through time and interact with each other to produce increasingly complex organized patterns.

The information content of the universe, in the Shannon sense, is the logarithm of the number of distinct histories that are consistent with the universe's observed macrostate. This is not a property of the universe's current configuration alone but of the entire trajectory from the initial state to the present: the measure of how many different ways the universe could have arrived at its current macrostate, given the constraints imposed by its physical laws and its initial conditions. The universe's rich structure — its galaxies, stars, planets, biospheres, and minds — represents an enormous reduction of this information from the initial state's maximum: the progressive elimination, through the formation of structure, of the trajectories that would have produced a featureless, high-entropy universe. Structure and information are duals: the formation of structure is the consumption of information, the production of organized patterns from the space of all possible patterns by the action of admissibility constraints.

13.7 Self-Organizing Systems

A self-organizing system is a system whose admissibility manifold has the property that locally admissible transitions tend to produce states with higher global admissibility: states from which a larger number of further strongly admissible reductions are possible, not fewer. This is the opposite of the spontaneous increase of entropy in isolated thermodynamic systems, and it requires an energy source — a thermodynamic gradient

— to maintain. Self-organization is not a violation of thermodynamics; it is a thermodynamically driven process in which the local dissipation of free energy is channeled by admissibility constraints into the production of global structure.

The formal structure of self-organization in the Spherepop framework is clear. A self-organizing system is an open system — one with an energy input — whose admissibility manifold has the property that the gradient of the action functional $\nabla S[\gamma]$ points, on average, toward states of higher structural complexity rather than lower. The energy input drives the system along this gradient, against the thermodynamic tendency toward high-entropy, low-structure states, producing and maintaining organized patterns that would spontaneously decay in the absence of the energy input. The organized pattern is an attractor of the constraint-navigation dynamics: a region of the admissibility manifold from which the local admissibility conditions tend to return the system, after small perturbations, to the organized state.

13.8 Agency and Thermodynamic Gradients

Agency — the capacity of a system to act in pursuit of goals, to select among available actions on the basis of anticipated consequences — is, on the Spherepop account, a thermodynamic gradient phenomenon: a property that emerges in systems that maintain far-from-equilibrium organization by dissipating thermodynamic gradients, and that has been shaped by evolution to exploit the admissibility structure of the environment in the service of the organism's survival and reproduction.

An agent, in this account, is a system that models its admis-

sibility manifold — that maintains an internal representation of which transitions are available from its current state, what their costs are, and what terminal states they lead to — and uses this model to select transitions that are likely to preserve or extend the organism's viability. Agency is therefore not a mysterious addition to the physical description of the organism; it is the functional name for the operation of a sufficiently accurate internal model of the admissibility manifold, used to guide trajectory selection in the service of maintaining the thermodynamic gradient that sustains the organism's organization.

The connection to the account of intelligence developed in Part II is direct. Intelligence, understood as predictive navigation through constraint space, is the cognitive form of agency: the capacity to model the admissibility manifold of the agent's environment accurately enough to select actions that lead to viable futures. Agency in simple organisms is a limited form of this capacity, operating over short time horizons and narrow environmental domains. Agency in cognitively complex organisms extends over longer time horizons and wider domains, exploiting the capacity for simulation and recursive representation developed in Chapter 7. Agency in social and institutional systems extends to the collective level, involving the distributed modeling of social admissibility manifolds by networks of agents acting in coordination.

13.9 The Universe as Constraint Navigation

The view of the universe as a constraint navigation process — as a trajectory through the space of admissible physical configurations, governed by the admissibility conditions encoded

in physical law and the action principle — is the cosmological generalization of the Spherepop framework's central claim. Physical law, on this view, is not a set of rules imposed on reality from outside; it is the admissibility structure of physical reality: the specification of which transitions among physical states are locally admissible, and which action functional governs the selection of globally admissible trajectories from the space of locally admissible ones.

The universe's history is a history in the Spherepop sense: an ordered sequence of admissible transitions through the space of physical configurations, from the initial state to the present, carrying provenance information — the specific sequence of transitions that occurred — that is not recoverable from the current state alone. The current state of the universe underdetermines its history, just as the terminal value of a computation underdetermines the trajectory that produced it. Understanding the universe's history requires access to the history itself, not merely to the current configuration — which is why cosmology is a historical science, and why the provenance of cosmic structures (the processes by which stars, galaxies, and planets formed) is scientifically important even for questions about the current universe's properties.

Chapter 14

Intelligence and Evolution

Five breakthroughs of intelligence are identified and examined: reinforcement and prediction, simulation and world modeling, language and recursive abstraction, computation as evolutionary process, and adaptive compression. The emergence of semantic layers and the topological stabilization of learning are treated as the culminating developments of this evolutionary sequence.

14.1 The Five Breakthroughs of Intelligence

The evolution of intelligence can be analyzed as a sequence of qualitative transitions — breakthroughs — in the capacity of biological systems to model and navigate their admissibility manifolds. Each breakthrough represents a new class of internal modeling capacity: a new way of representing, compressing, and exploiting the constraint structure of the environment in the service of the organism's viability. The five breakthroughs identified here are not exhaustive, but they mark the major transitions in the evolutionary history of intelligence that are relevant to the concerns of this monograph.

The first breakthrough is *reactive sensing*: the capacity to detect features of the current environment that are relevant to admissibility and to trigger locally appropriate responses. The second is *predictive modeling*: the capacity to represent future states of the environment and to select actions on the basis of predicted consequences rather than only current stimuli. The third is *social modeling*: the capacity to represent other agents as systems with internal states — beliefs, intentions, goals — and to predict their behavior by modeling those states. The fourth is *linguistic abstraction*: the capacity to represent and communicate admissibility structures in symbolic form, decoupled from immediate sensory experience and transmissible across individuals and generations. The fifth is *computational recursion*: the capacity to reason about reasoning itself, to model models, and to construct formal systems whose admissibility conditions are explicitly specified rather than implicitly embodied in biological structure.

Each breakthrough enables new forms of constraint navigation that were unavailable at the previous level, and each produces new forms of cognitive organization whose properties are not predictable from the properties of the preceding level. The sequence is cumulative: each later breakthrough builds on the capacities established by the earlier ones, producing increasingly powerful and flexible forms of admissibility modeling.

14.2 Reinforcement and Prediction

The first two breakthroughs — reactive sensing and predictive modeling — are closely linked in the evolutionary history

of intelligence and in the computational framework of modern reinforcement learning. A reinforcement learning agent learns a policy: a mapping from perceived states to actions, optimized to maximize expected cumulative reward over time. The learning process is a form of constraint navigation: the agent explores the admissibility manifold of its environment — learning which actions are available from which states, what their consequences are, and what rewards they produce — and updates its policy to select actions whose long-run consequences are most rewarding.

The transition from reactive sensing to predictive modeling corresponds, in the reinforcement learning framework, to the transition from model-free to model-based reinforcement learning. A model-free agent learns a direct mapping from states to values or actions, without constructing an explicit model of the environment's transition dynamics. A model-based agent constructs an explicit model — a representation of the admissibility manifold: which transitions are available from which states, with what probabilities and what rewards — and uses this model to plan actions by simulating their consequences before committing to them. The model-based agent is more data-efficient, more flexible, and more capable of generalization, precisely because it has access to an explicit representation of the constraint structure rather than only an implicit summary encoded in the learned policy.

14.3 Simulation and World Modeling

The capacity for mental simulation — the internal generation of representations of hypothetical world states, using a model of

the environment's dynamics — is the cognitive foundation of predictive intelligence, and it is the capacity that most directly implements the Spherepop framework's picture of intelligence as admissibility manifold navigation. A system that can simulate the consequences of actions before committing to them is a system that can explore its admissibility manifold internally, without incurring the real-world costs of physical exploration. The quality of its navigation is limited by the accuracy of the model used for simulation, which is determined by the history of prior interactions with the environment and the capacity of the modeling system to extract and compress the relevant constraint structure.

The relationship between simulation capacity and the action formalism of Chapter 12 is direct. A system that simulates the consequences of a candidate action is computing, in effect, the action cost $S[\gamma]$ of the trajectory that begins with that action: it is evaluating the expected change in accessibility volume and structural commitment that the action would produce. The principle of least structural commitment — the selection of the trajectory with stationary action — is, from the perspective of the simulating agent, the selection of the action whose simulated consequence trajectory has the lowest total cost. Intelligence, in this sense, is the capacity to evaluate the action functional of candidate trajectories before committing to any of them.

14.4 Language and Recursive Abstraction

Language is the medium through which admissibility structures are made publicly accessible and transmissible across indi-

viduals and generations. A language is a system of expressions whose admissibility conditions are shared among the users of the language: the semantic conventions of the language specify which transitions among semantic states are locally admissible, which expressions are compatible with which contexts, and which inferences from given premises are sanctioned by the language's semantic norms. Understanding a language is having an accurate internal model of its admissibility manifold: the capacity to predict which expressions are well-formed, which inferences are valid, and which contexts are appropriate for which expressions.

The recursive abstraction enabled by language is the capacity to use linguistic expressions to describe the admissibility conditions of other linguistic expressions: to speak about meaning, to formulate rules, to construct proofs, and to reason about reasoning. This recursive capacity is what distinguishes language from other signaling systems: it enables not merely the communication of information about the environment but the communication and mutual refinement of the admissibility structures through which the environment is conceptualized. Science, mathematics, philosophy, and law are all practices organized around the collective refinement of admissibility structures through linguistic exchange — through the mutual testing, criticism, and revision of claimed admissibility conditions in public discourse.

14.5 Computation as Evolutionary Process

The formal theory of computation has its own evolutionary structure, paralleling the biological evolution described in the

preceding sections. Programming languages evolve: they accumulate new features, new type systems, new constraint mechanisms, as the community of their users identifies new admissibility conditions that are useful and new constraints that improve the reliability and intelligibility of programs. Programming paradigms evolve: the shift from imperative to functional to declarative to logic programming represents a sequence of transitions in which the admissibility conditions governing computation become more explicit and more structurally sophisticated.

The Spherepop framework participates in this evolutionary process. It represents a specific direction of evolution: the direction of making evaluation history a first-class object, of externalizing admissibility conditions into the visible structure of the representation, and of treating the trajectory through the admissibility manifold as the primary semantic object rather than the terminal value it produces. This direction is motivated by the same evolutionary pressure that has driven the development of type systems, formal verification tools, and proof assistants: the pressure to make the admissibility conditions of computation explicit and verifiable rather than implicit and hoped-for.

14.6 Adaptive Compression

The capacity for adaptive compression — the formation of compressed representations that preserve the structural invariants relevant to the organism's current needs, and that can be revised when those needs change — is the cognitive form of the collapse operation formalized in the Spherepop framework.

Learning is, at its core, the process of forming compressed models of admissibility manifolds: representations that are smaller than the full manifold but that support accurate navigation of the most frequently traversed regions.

Adaptive compression differs from fixed compression in the same way that understanding differs from memorization: the compressed model is not a static lookup table but a dynamic structure whose compression parameters adapt to the organism's experience. When a new region of the admissibility manifold is explored — when the organism encounters a situation its model does not cover — the model is updated to extend its coverage, revising the compression in a way that preserves the structural invariants that have proven useful while accommodating the new information. This update process is governed by the same principle as the broader Spherepop formalism: it must be a strongly admissible reduction of the model, preserving its structural coherence while reducing its mismatch with the encountered environment.

14.7 The Emergence of Semantic Layers

The evolution of language and recursive abstraction produces a distinctive organizational feature: the emergence of semantic layers — levels of description that are each coherent and relatively autonomous, with their own admissibility conditions, but that are coupled to each other through systematic mappings that allow descriptions at one level to be grounded in, or translated into, descriptions at another level. The semantic layers of a developed language include the phonological level, the morphological level, the syntactic level, the semantic level,

and the pragmatic level. Each has its own admissibility conditions; each is coupled to the adjacent levels by systematic correspondence rules.

The emergence of semantic layers is an instance of the general phenomenon of nested admissibility hierarchies: each layer's admissibility conditions are constraints on the transitions available at that layer, and the coupling between layers is a system of inter-level constraints that propagates admissibility conditions from one layer to another. The Spherepop framework describes this coupling naturally through the nested bubble structure: the admissibility conditions of an outer bubble (a higher-level semantic description) constrain the admissibility conditions of the inner bubbles (the lower-level realization), and the pop results of inner bubbles propagate upward to affect the admissibility conditions of the outer bubble.

14.8 Learning as Topological Stabilization

Learning, in the account developed across Part II and this chapter, is the process by which an organism's internal model of the admissibility manifold of its environment becomes more accurate through interaction with that environment. The process has a specific topological character: it is not the mere accumulation of data points but the progressive stabilization of the model's topological structure — the convergence of the model's topology toward the actual topology of the admissibility manifold, measured by the degree to which the model correctly predicts which transitions are locally admissible.

Topological stabilization means that as learning proceeds, the model's admissibility conditions become more reliably con-

tinuous with the actual admissibility conditions: small changes in the environment produce small changes in the model's predictions, and the model generalizes correctly to nearby regions of the admissibility manifold that it has not directly explored. A model that has achieved topological stabilization in a domain supports reliable navigation in that domain: it can be used to plan novel trajectories through previously unexplored regions of the manifold and to predict the consequences of actions in new situations.

14.9 The Growth of Internal Models

The growth of internal models — the progressive increase in the complexity, accuracy, and scope of the models that intelligent systems maintain of their admissibility manifolds — is the long-term trajectory of cognitive evolution, visible across the evolutionary history of intelligence, the developmental history of individual organisms, and the cultural history of human knowledge. In each case, the trajectory is not simply from less to more; it is from shallower to deeper structural organization, from models that accurately describe a narrow domain to models that accurately describe a wider domain through a richer and more explicitly structured representation of the admissibility conditions governing it.

The growth of internal models is constrained by the same principles that govern the growth of any complex self-organizing system. The model must be strongly admissible in the sense of Section 1.5: it must maintain structural coherence as it grows, preserving transportable invariants across revisions, and it must maintain path-sensitivity — the accumulated history of

the model's development must genuinely modify its future admissibility conditions, producing a model that is richer and more structurally organized than any model that could have been produced by a different developmental history. The internal model of an experienced practitioner in a complex domain is not merely a model with more entries in its lookup table; it is a model with a deeper and more structurally organized admissibility manifold, shaped by the specific history of encounters, revisions, and integrations that constitute expertise.

Part V

Critiques of Rationalism and Technique

Chapter 15

Feyerabend and the Limits of Method

Feyerabend's challenge to universal rational procedure is examined seriously and without caricature. The historical messiness of scientific discovery is used to motivate a conception of science as evolving practice rather than algorithmic execution. "Anything goes" is revisited and its genuine insight separated from its overreach.

15.1 Against Universal Rational Procedure

Paul Feyerabend's *Against Method* is the most sustained and provocative philosophical attack on the idea that science proceeds, or should proceed, by following a universal rational method — a set of explicit, context-independent rules for the generation and evaluation of scientific hypotheses. Feyerabend's argument is historical: he examines the actual history of scientific practice and finds that the episodes most celebrated as triumphs of scientific reason — the Copernican revolution, Galileo's mechanics, the development of quantum theory —

were achieved not by following the official methodology of their time but by violating it: by ignoring inconvenient evidence, by borrowing arguments from philosophy and rhetoric, by exploiting social and political circumstances, by persisting with theories that were refuted by the available data.

The force of Feyerabend's argument is directed not at science as such but at the specific claim that there is a universal rational procedure — a method — that distinguishes science from non-science and that, if followed, reliably produces scientific progress. This claim, which Feyerabend attributes to Popper's falsificationism and to the logical empiricist tradition more broadly, he finds both historically false and methodologically harmful: historically false because science has never actually proceeded by universal method, and methodologically harmful because insisting on universal method suppresses the kind of theoretical diversity and rule-breaking that historically drove the most important scientific innovations.

15.2 The Historical Messiness of Discovery

The historical case studies that Feyerabend marshals in support of his argument reveal a consistent pattern: at the moments of greatest scientific creativity, the scientists involved were not following the officially endorsed rules of scientific method but were constructing local, ad hoc, context-specific strategies for advancing their particular research programs. Galileo's argument for Copernicanism involved appeals to unverified theoretical principles, the use of instruments whose reliability was contested, the deployment of rhetorical techniques borrowed from the tradition of humanist persuasion, and the persistent

dismissal of empirical observations — such as the behavior of falling bodies and the absence of observable stellar parallax — that constituted genuine evidence against the Copernican theory as then formulated.

This historical messiness is not, for Feyerabend, evidence of Galileo's irrationality or dishonesty. It is evidence that scientific progress requires the kind of flexibility, opportunism, and rule-violation that a universal rational procedure would forbid. The rules of method are codifications of past successful practice, and they are therefore reliable guides for producing more of the same. But the most important scientific advances are not more of the same; they are departures from established practice that open new domains and new methods. A methodology that rules out such departures in advance is a methodology that makes scientific stagnation a virtue.

15.3 Science as Evolving Practice

The Spherepop framework offers a perspective on Feyerabend's historical argument that is sympathetic to its substance while more precise about its implications. Science, on the Spherepop account, is an evolving system of admissibility conditions: a practice whose admissibility conditions — what counts as a legitimate hypothesis, what counts as acceptable evidence, what inferential moves are sanctioned — are not fixed in advance but are themselves subject to revision through the practice. The history of science is a history of constraint formation and revision: of the accumulation of new admissibility conditions through successful practice, and of the occasional revision of established conditions when they prove incompatible with new

evidence or new theoretical possibilities.

This account vindicates Feyerabend's historical observation without endorsing his epistemological nihilism. The absence of a universal rational procedure does not mean that science is merely rhetoric or politics, or that all scientific claims are equally valid. It means that the admissibility conditions of scientific practice are historically accumulated and contextually specific, rather than context-independent and fixed in advance. The admissibility conditions of contemporary physics are different from those of Galileo's time, and rightly so: they incorporate what has been learned from three centuries of physical inquiry about what kinds of evidence, argument, and theoretical structure reliably produce accurate models of physical reality.

15.4 "Anything Goes" Revisited

Feyerabend's provocative slogan — "anything goes" — has been widely misread as an endorsement of epistemological relativism: the view that any cognitive practice is as good as any other, that there are no grounds for preferring one theory over another, and that science is merely one of many equally valid ways of understanding the world. This reading misses the actual target of the slogan, which is not epistemic standards in general but universal rational procedure in particular.

What Feyerabend means by "anything goes" is the much more modest claim that no fixed set of methodological rules is sufficient to characterize successful scientific practice across all domains and all historical periods. At any given moment, the scientist who is constrained only by the official rules of method

is less capable than the scientist who can also draw on non-standard strategies, theoretical innovations, and rule-violating moves when the occasion demands. The slogan is a counsel of flexibility, not of nihilism: it says that the admissibility conditions of scientific practice should not be so rigid that they exclude the kinds of theoretical creativity that drive the most important advances.

Read in this way, "anything goes" is not in tension with the Spherepop framework but is continuous with it. The framework's insistence that admissibility conditions are contextually specific rather than universally fixed is precisely Feyerabend's point. But the framework also insists that there are admissibility conditions: that not every cognitive move is equally valid, that the evaluation of scientific claims is governed by real constraints, and that the constraints change through the history of science without becoming merely arbitrary. Against Feyerabend's tendency toward epistemological free-for-all, the Spherepop framework insists on the reality and importance of admissibility — while agreeing with Feyerabend that the specific content of admissibility conditions is historically accumulated rather than universally fixed.

Chapter 16

Popper and Rational Criticism

Popper's framework of rational criticism and error correction is defended against Feyerabend's critique, while its limitations are acknowledged. Falsifiability is treated not as a demarcation criterion but as a structural commitment to dialogic openness.

16.1 Frameworks and Dialogue

Karl Popper's philosophy of science, developed most fully in *The Logic of Scientific Discovery* and *Conjectures and Refutations*, is organized around a conception of rational inquiry as dialogic: as the public exchange of conjectures and criticisms, governed by the norm that each participant is genuinely open to having their views refuted by evidence or argument. The Popperian rational dialogue is not a procedure for discovering truth; it is a procedure for eliminating error. By subjecting conjectures to the most severe available criticism, the dialogue eliminates theories that cannot survive critical scrutiny, leaving the sur-

viving theories provisionally accepted but permanently subject to further challenge.

The dialogic structure of Popperian inquiry maps directly onto the Spherepop framework's treatment of admissibility as a collective and revisable structure. The admissibility conditions of scientific discourse — what counts as a legitimate conjecture, what counts as a genuine refutation, what inferential moves are sanctioned in the scientific dialogue — are maintained by the community of practitioners through the ongoing practice of mutual criticism. They are not fixed by any authority but maintained and revised through the very dialogic process they govern. The admissibility conditions of science are self-refining: the practice of scientific dialogue is itself subject to criticism and revision, so that the conditions governing the dialogue can be improved through the dialogue they govern.

16.2 Error Correction and Falsifiability

Falsifiability, in Popper's account, is the criterion that distinguishes genuinely scientific claims from pseudo-scientific ones: a claim is scientific if and only if it is possible, in principle, to specify observations that would refute it. This criterion has been extensively criticized — for excluding claims that are intuitively scientific (probabilistic claims, historical claims), for including claims that are intuitively pseudo-scientific (astrology with sufficiently vague predictions), and for resting on a naïve account of the relationship between theory and evidence that ignores the Duhem-Quine underdetermination problem.

The Spherepop framework offers a reconstruction of falsifiability that addresses these criticisms. Falsifiability, recon-

structed in Spherepop terms, is not a binary property of individual claims but a structural property of the admissibility conditions governing those claims: a claim is falsifiable if its admissibility conditions include the condition that specific observable events would, if they occurred, constitute locally inadmissible transitions in the theory's admissibility manifold — events that the theory would need to categorically refuse. This reconstruction is more flexible than Popper's original version: it allows for probabilistic falsifiability (observations that sufficiently reduce the theory's global admissibility score), historical falsifiability (evidence from the past that is in principle accessible), and theory-laden falsifiability (the recognition that what counts as a refuting observation depends on the background theory). But it preserves the core Popperian insight: a scientific claim must be structured in a way that makes it genuinely responsive to evidence — that makes some possible observations admissibility-reducing — rather than compatible with any evidence whatsoever.

16.3 The Defense of Rational Inquiry

Against Feyerabend's pluralism, Popper defends the reality and importance of rational standards: the claim that some ways of forming and revising beliefs are better than others, that the evaluation of theories is not merely a matter of social convention or rhetorical persuasion, and that the progress of science represents a genuine improvement in our understanding of the world rather than merely a change in fashion. This defense is not a defense of any specific methodology as universally correct; it is a defense of the norm of rational criticism itself — the

requirement that claims be subjected to genuine scrutiny and that the outcome of that scrutiny be taken seriously.

The Spherepop framework supports this defense at the structural level. Rational inquiry, on the Spherepop account, is inquiry that is responsive to its admissibility conditions: that takes seriously the constraints imposed by evidence, logical consistency, and theoretical coherence, and that revises its claims when those constraints are violated. The rationality of scientific inquiry is not a matter of following fixed rules but of maintaining genuine responsiveness to the admissibility conditions of the community's evolving epistemic practice. A community that ignores refuting evidence, that tolerates logical inconsistency, or that fails to revise its claims in response to genuine criticism is a community whose inquiry has become inadmissible — not because it violates a fixed methodological rule, but because it has ceased to be genuinely constrained by the evidence and argument that constitute the admissibility conditions of honest inquiry.

16.4 Structural Critique Without Dogmatism

The position that emerges from the dialogue between Feyerabend and Popper — and that the Spherepop framework articulates at a more formal level — is what might be called structural critique without dogmatism: the commitment to maintaining genuine admissibility conditions for cognitive practice, while acknowledging that the specific content of those conditions is historically accumulated, contextually specific, and permanently subject to revision.

Structural critique differs from both dogmatic rationalism

and epistemological relativism. Dogmatic rationalism holds that there are fixed, universal rules of rationality that determine in advance what is admissible in inquiry. Epistemological relativism holds that there are no admissibility conditions beyond social convention: that any cognitive practice is as valid as any other. Structural critique holds that admissibility conditions are real and important — that they constitute the constraint structure of honest inquiry — while insisting that their specific content is itself a product of the historical practice they govern, not a fixed set of rules determined in advance.

The Spherepop formalism captures this position precisely. The admissibility manifold of a cognitive practice is real: it constrains which transitions are locally admissible, and these constraints are not merely conventions. But the admissibility manifold is also historical: it has been shaped by the accumulated history of the practice, and it continues to evolve through the practice. The admissibility conditions of contemporary physics are more refined, more accurate, and more structurally sophisticated than those of Galileo's time, because they incorporate the lessons of three centuries of physical inquiry. This evolution is not arbitrary; it is itself constrained by the admissibility conditions in force at each stage. The history of science is a strongly admissible sequence of reductions of the admissibility manifold of physical inquiry.

Chapter 17

Between Feyerabend and Popper: Spherepop as Non-Dogmatic Rationalism

Spherepop is positioned in the intellectual space between Feyerabend's pluralism and Popper's rationalism. The chapter argues that Spherepop is not anti-rational but anti-opaque: interpretability is a philosophical principle, and visible structure is the condition of possibility for genuine public reason.

17.1 Why Spherepop Is Not Anti-Rational

Spherepop might appear, at first encounter, to be a form of anti-rationalism: a rejection of formal symbolic systems in favor of spatial intuition, a privileging of visual cognition over logical rigor, a rehabilitation of pre-formal ways of thinking that the development of mathematics and logic was supposed to have superseded. This appearance is misleading. The critique Spherepop advances is not a critique of rationality but a critique of a specific historical form that rationality has taken: the form

of linear symbolic compression that optimizes representational density at the cost of visible structural provenance.

The distinction is crucial and must be stated with care. Spherepop does not reject formal rigor; it extends formal rigor to include the history of the computation as a formal object, the admissibility conditions as explicit structural constraints, and the collapse operations as operations with precise mathematical specifications. It does not reject abstraction; it insists that abstraction be performed in ways that preserve the structural invariants necessary for further admissible reasoning. It does not reject symbolic notation; it argues that symbolic notation should be supplemented or replaced, in domains where visible structure matters for understanding, by representational systems that externalize the structural information that symbols compress away. The objection is not to symbolism but to the specific choice of what to compress and what to preserve.

17.2 Against Opacity, Not Against Structure

The target of the Spherepop critique is opacity: the concealment of structural information that is operationally relevant behind notational conventions, implementation details, or representational choices that make the information difficult or impossible to recover from the surface form of the representation. Opacity is not the same as abstraction. Abstraction is the selective retention of structurally relevant information and the discarding of structurally irrelevant detail. Opacity is the discarding of structurally relevant information along with the irrelevant detail, producing a representation that is formally adequate within a limited domain but that fails to support the reasoning

and generalization that the full structural information would enable.

The distinction between abstraction and opacity is precisely the distinction between structured compression (a semantically valid collapse) and degenerative collapse (a semantically invalid one). Abstraction is collapse that preserves transportable invariants; opacity is collapse that destroys them. The calculator that returns 13 has abstracted from the specific intermediate computational steps — which is appropriate if all one needs is the number — but has also collapsed the evaluation order, the scope structure, the admissibility profile, and the provenance of the result, which is opacity if one needs those things and cannot recover them. Whether a given collapse is abstraction or opacity depends entirely on what the collapsed information is needed for: there is no absolute answer, only contextual ones indexed to the relevant admissibility conditions.

17.3 Interpretability as a Philosophical Principle

Interpretability — the property of a system of being understandable by the agents who must work with it — is not merely a practical convenience but a philosophical principle that follows from the deeper commitments of the Spherepop framework. If understanding is stabilized navigational compression of the admissibility manifold, then an interpretable system is one whose admissibility manifold is exposed to its users in a form that supports stable navigation: a form in which the constraint conditions governing the system's behavior are visible, the evaluation history is preserved, and the structural invariants that determine what the system will do next are accessible to the

reasoning of the people who must rely on it.

An uninterpretable system — a system whose admissibility manifold is hidden behind an opaque interface that exposes only terminal outputs — is one that its users cannot genuinely understand in the Spherepop sense. They can navigate it in the weakest sense: they can learn, by trial and error, which inputs produce which outputs. But they cannot navigate it in the sense of understanding it: they cannot form an accurate model of its admissibility conditions, cannot predict its behavior in novel situations by reasoning about its constraint structure, and cannot diagnose errors by tracing them back through the computation's history. Their relationship to the system is that of a calculator user to the calculator: they can operate it but they cannot understand it.

The philosophical principle is this: systems on which significant decisions depend — systems whose outputs govern individual life choices, institutional decisions, or public policy — must be interpretable by the people who depend on them, not merely operable. The opacity of such systems is not merely a technical inconvenience but a structural violation of the epistemic conditions necessary for genuine agency: the capacity to understand and therefore to genuinely authorize or contest the decisions being made on one's behalf.

17.4 Visible Structure and Public Reason

The connection between visible structure and public reason is the political dimension of the Spherepop framework's philosophical commitments. Public reason, in the liberal political tradition, is the kind of reasoning that can be offered to and

assessed by all members of a political community, regardless of their particular comprehensive doctrines or personal interests. It is reasoning that is public in the sense of being conducted in terms that all can evaluate and challenge, rather than private in the sense of being accessible only to specialists, or opaque in the sense of being embedded in algorithmic processes that no one can fully inspect.

Visible structure is the condition of possibility for public reason in the domain of computation and technical decision-making. A technical system whose structural operation is visible — whose admissibility conditions, evaluation history, and constraint structure are accessible to the people affected by its outputs — is a system that can be incorporated into public reason: its outputs can be publicly contested, its admissibility conditions publicly debated, its failure modes publicly diagnosed. A system whose operation is opaque — whose outputs are produced by processes that neither its users nor its developers can fully trace — is a system that operates outside public reason: its outputs cannot be genuinely contested because their structural basis is not publicly accessible.

The political stakes of computational opacity are not abstract. Opaque algorithmic systems increasingly govern credit decisions, criminal sentencing recommendations, hiring processes, content distribution, and medical diagnosis. The structural invisibility of these systems is not a technical limitation awaiting improvement; it is a political condition: the condition of having one's opportunities, penalties, and life chances shaped by processes one cannot understand, contest, or appeal. Spherepop's insistence on visible structure is, at its political dimension, an insistence on the conditions necessary for genuine

democratic accountability of technical systems.

Chapter 18

Ellul and Technique

Jacques Ellul's analysis of technique—standardized procedure in the service of efficiency at the cost of meaning—is applied to contemporary computational practice. The automation of thought and the erasure of semantic content are identified as the structural consequences of unreflective technique, and structural transparency is proposed as the antidote.

18.1 Technique as Standardized Procedure

Jacques Ellul's concept of *technique* — developed most fully in *The Technological Society* — refers not to technology in the narrow sense of machines and tools but to a broader phenomenon: the systematization of all human activity into standardized, efficient procedures optimized toward measurable outcomes. Technique, for Ellul, is the disposition to treat every domain of human life — work, education, medicine, art, religion, politics — as a technical problem requiring a technical solution: a solution consisting of optimized procedures, quantified metrics, and efficient processes, without residue of the qualitative, the particular, or the irreducibly personal.

Ellul's analysis is relevant to the Spherepop framework because technique, in his sense, is precisely the disposition to privilege terminal outputs over evaluation histories: to optimize for measurable outcomes — productivity, efficiency, accuracy, throughput — without regard for the structural processes that produce those outcomes or the admissibility conditions that govern them. Technique is the social and institutional form of the calculator's epistemology: the reduction of complex, historically constituted practices to output-producing procedures, with the elimination of the provenance, the judgment, and the structural understanding that the practices originally required.

18.2 Efficiency and the Erasure of Meaning

The drive for efficiency — the optimization of output per unit of input — is the primary mechanism by which technique erases meaning. Meaningful practices are typically inefficient: they involve the maintenance of historical continuity, the exercise of contextual judgment, the preservation of structural complexity that cannot be encoded in a procedure without loss. The craftsman who makes a chair expends more time and produces fewer chairs per hour than the factory worker following a standardized procedure, because the craftsman's practice involves attending to the specific properties of specific materials in specific contexts in ways that the standardized procedure cannot accommodate. The efficiency gain of the factory is real; but it is achieved by eliminating the structural specificity — the constraint-sensitive responsiveness to particulars — that constitutes the craft's meaning.

This is the pattern Ellul identifies across every domain

of social life that technique colonizes. The efficient hospital processes patients more quickly than the inefficient one, but achieves its efficiency by eliminating the physician's attention to the patient as a particular person with a particular history — by standardizing the diagnostic and therapeutic procedures to eliminate the contextual judgment that responsive medicine requires. The efficient school produces higher standardized test scores than the inefficient one, but achieves its efficiency by eliminating the teacher's attention to the particular developmental trajectory of the particular student — by standardizing the pedagogical procedures to produce measurable outputs without regard to the understanding that the outputs are supposed to indicate.

18.3 The Automation of Thought

The automation of thought — the replacement of human judgment by algorithmic procedure — is the limiting case of technique's colonization of cognitive practice. When the procedures that govern action are sufficiently complex and the outputs they optimize are sufficiently quantified, the procedures can be implemented in software and executed without human involvement. The automation is more efficient than the human version: it is faster, cheaper, more consistent, and immune to the particular forms of human error that arise from fatigue, distraction, and individual variation. But it also inherits the characteristic limitation of technique: it optimizes for the measurable output while eliminating the contextual judgment, the historical responsiveness, and the structural understanding that the practice originally involved.

The automated system is, in the Spherepop sense, a system that performs collapse without provenance: it maps inputs to outputs through a process that records no history, maintains no admissibility conditions that can be inspected, and produces no record of the structural basis for its outputs. It is efficient precisely because it has eliminated the overhead of maintaining visible structure. But it is also blind to the admissibility conditions that should govern its outputs: it cannot recognize the cases in which its standardized procedure produces an output that violates the structural constraints of the domain, because it has no representation of those constraints — only a learned mapping from inputs to outputs that approximates the behavior of the constrained practice in the training distribution and fails outside it.

18.4 Why Structural Transparency Matters

Structural transparency — the property of a system that makes its admissibility conditions, evaluation history, and constraint structure visible to the people who use and are affected by it — is the antidote to the epistemological and political pathologies of technique. It matters for four distinct reasons, each grounded in the framework developed across this monograph.

It matters *cognitively* because understanding requires access to the admissibility manifold: a user who cannot see the constraint structure of a system cannot understand it in the Spherepop sense — cannot form a stable, generalization-supporting model of what the system will do in novel situations. They can operate the system but not understand it, producing the characteristic fragility of expertise built on technique: competence

within the training distribution, failure at the boundaries.

It matters *epistemically* because genuine knowledge of a domain requires structural transparency of the representations used to reason about it. A practitioner whose representations are opaque — whose conclusions are outputs of procedures they cannot inspect — cannot engage in the kind of critical self-reflection and error correction that genuine expertise requires. They cannot trace errors back to their structural source, cannot identify the admissibility conditions that their practice is violating, and cannot revise their practice in structurally informed ways.

It matters *politically* because democratic accountability requires that the processes governing public decisions be assessable by the public: that the admissibility conditions of technical decision-making be exposed to contestation and revision through public discourse. An opaque system that governs public decisions is a system that operates outside the scope of democratic accountability, producing outcomes that cannot be genuinely contested because their structural basis is inaccessible.

It matters *ethically* because the reduction of human beings to inputs in an output-optimization procedure — the elimination of their particular histories, contextual needs, and irreducible specificities in favor of the standardized treatment appropriate for their output-relevant features — is a form of epistemic injustice: a failure to treat them as beings whose full structural particularity is morally relevant and deserving of responsive, history-sensitive attention. Structural transparency is the formal condition for the kind of attention that ethical treatment of persons requires.

Part VI

Ontology, Language, and Explainability

Chapter 19

Symbolic Versus Subsymbolic Systems

The rise of statistical models and the crisis of semantic grounding are examined. Large language models are analyzed as instances of opaque compression: powerful, but constitutively resistant to explainability in the absence of additional structural scaffolding. Explainability and interpretability are distinguished with precision.

19.1 The Rise of Statistical Models

The twentieth century witnessed two broadly distinct traditions in the formal treatment of language and meaning. The symbolic tradition, descending from Frege, Russell, and the early Wittgenstein through the formal semantics of Montague and the proof-theoretic approaches of Gentzen and Martin-Löf, treated meaning as compositionally structured: the meaning of a complex expression was determined by the meanings of its parts and the rules governing their combination. Sentences

had truth conditions, expressions had denotations, and the semantic relation between language and world was mediated by an explicit compositional calculus whose operations were individually interpretable and collectively traceable. The computational tradition that emerged from this lineage — symbolic AI, expert systems, knowledge representation, logic programming — inherited the same commitments: knowledge was encoded in explicit propositional form, inference was rule-governed, and the reasoning process was in principle fully traceable from premises to conclusion.

The statistical tradition has a different genealogy. Its roots lie in the frequentist and Bayesian traditions in probability theory, in the information-theoretic work of Shannon, in the connectionist movement of the 1980s, and ultimately in the neural network architectures that have come to dominate machine learning. Where the symbolic tradition treats meaning as explicitly structured, the statistical tradition treats it as implicitly encoded in patterns of co-occurrence, distributional proximity, and learned numerical weights. A statistical language model does not represent the meaning of a word by listing its semantic relations to other words in a structured knowledge base. It represents meaning as a position in a high-dimensional numerical space whose geometry is learned from patterns of usage in a training corpus. The relations among meanings are encoded in the geometry of this space — semantically similar words occupy nearby positions — but the geometry is not constructed according to explicit semantic rules. It emerges from optimization of a numerical objective over vast quantities of text.

The rise of statistical models represents an enormous prac-

tical advance. Systems trained on large corpora learn semantic regularities — analogical relations, taxonomic structure, contextual appropriateness — that symbolic systems struggled to capture without enormous manual effort. They generalize across surface forms, tolerate ambiguity and noise, and scale in ways that symbolic approaches resisted. These are real achievements. But they come with a structural cost that the symbolic tradition did not pay: the learned representations are distributed across millions of numerical parameters, with no individual parameter or small group of parameters carrying interpretable semantic content. The geometry of the semantic space encodes meaning, but the encoding is opaque — not because the space is unavailable for inspection, but because the relation between geometric position and semantic content is not expressed in a form that human cognition can read directly.

19.2 Language Models and Opaque Compression

A large language model is, from the perspective of the Spherepop framework, a compressed representation of an admissibility manifold: specifically, the admissibility manifold of natural language use, encoding which linguistic continuations are contextually admissible given a particular preceding context. The compression is achieved through the learned numerical parameters of the model, which collectively approximate the conditional probability distributions $P(\text{next token} \mid \text{context})$ over the training distribution. This is a form of compression in the strict technical sense: a large and complex object — the distribution over all possible linguistic continuations — is represented by a finite set of numerical parameters whose collective

behavior approximates the original distribution.

The compression is, by the standards of terminal-output accuracy, highly effective. Language models produce contextually appropriate continuations across an enormous range of domains, styles, and registers, and they do so without explicit representation of the semantic or syntactic rules governing appropriateness. The admissibility manifold of language use has been approximated, with remarkable fidelity in many regions, by a purely statistical model.

But the compression is also, by the standards of the Spherepop framework, a form of opacity. The model's parameters do not represent admissibility conditions in a form that is accessible to the Principle of Contextual Equivalence: one cannot inspect the model's weights and determine which equivalence relation is being applied when two different input contexts produce the same output distribution. The model does not expose its collapse structure. It does not distinguish between histories that are genuinely semantically equivalent under some principled admissibility criterion and histories that produce identical outputs for accidental statistical reasons. And it does not preserve provenance: when the model generates a continuation, the generation is not accompanied by a record of which constraints governed the generation, which alternatives were considered and rejected, or which equivalence classes of input context the model is treating as interchangeable.

This is not a criticism of large language models as useful tools. It is a precise characterization of what they are, stated in the vocabulary of the Spherepop framework. They are effective admissibility approximators whose collapse structure is not exposed. Whether this opacity matters depends on what one

needs from the system. For many practical tasks — drafting text, answering factual questions, summarizing documents — the opacity is irrelevant: the terminal output is sufficient. For tasks that require understanding of the reasoning process, verification of the grounds for a claim, or detection of the equivalence relation being implicitly applied, the opacity is a fundamental limitation that no amount of increased model scale can resolve, because the limitation is architectural rather than quantitative.

19.3 Explainability and Interpretability

The terms *explainability* and *interpretability* are frequently used interchangeably in the literature on machine learning systems, but the Spherepop framework provides a principled basis for distinguishing them, and the distinction matters enormously for assessing the prospects and limitations of current approaches to AI transparency.

Explainability is the capacity to generate, post-hoc, a human-readable account of why a system produced a particular output in a particular case. An explainable system can produce saliency maps showing which input features influenced the output, decision trees approximating local behavior, or natural-language rationales for specific decisions. Explanations are generated after the fact and need not correspond to the actual computational process the system executed. They are best understood as compressed summaries of the output distribution in the neighborhood of a specific input — approximations to the local structure of the admissibility manifold, generated on demand, without any guarantee that the approximation is faithful.

Interpretability is the stronger property: a system is interpretable if its computational process is expressed in a form that allows the admissibility structure — the constraint configuration governing each step — to be read directly from the system's representation. An interpretable system exposes its collapse structure: one can inspect it and determine which equivalence relation governs each step, which alternatives were locally admissible and why some were excluded, what the commitment cost of each transition was, and how the terminal output follows from the specific sequence of admissible transitions that produced it. Interpretability is not a post-hoc property; it is a structural property of the representation itself.

Definition 19.1. A computational system $\mathcal{S}$ is *interpretable* (in the Spherepop sense) if its representation exposes, for each step e_t of each trajectory γ: the constraint configuration C_t in force at step t; the local admissibility condition $\mathsf{Adm}_{\text{local}}(e_t \mid C_t)$; the action cost $\mathcal{L}_t$; and the updated constraint configuration C_{t+1} produced by e_t. A system that generates post-hoc accounts of its behavior without exposing this structure is *explainable* but not interpretable.

The distinction has immediate practical implications. An explainable system can be audited case by case, with uncertain fidelity, at the cost of substantial additional computation. An interpretable system can be audited structurally, by reading its representation directly, at the cost of designing the system's representation to be readable in the first place. Current large language models are, in varying degrees, explainable but not interpretable. Spherepop-style systems, if built, would be interpretable by construction: the evaluation history is part of the

representation, and the constraint configuration at each step is directly readable from the bubble topology.

19.4 The Crisis of Semantic Grounding

The problem of semantic grounding asks: how does a formal symbol system acquire genuine reference to the world? The question was posed most sharply by Searle's Chinese Room argument and by Harnad's symbol grounding problem, both of which argued that formal symbol manipulation is insufficient for genuine meaning because symbols derive their meaning from their connections to sensorimotor experience, not from their relations to other symbols. A system that manipulates symbols correctly according to syntactic rules is not thereby understanding those symbols; it is processing formal marks according to formal rules, and the question of what those marks mean is not answered by describing the rules.

Statistical language models appear, superficially, to escape this critique because their representations are not purely syntactic: they encode semantic regularities learned from patterns of use, and those regularities reflect genuine structural features of the domains their training corpora describe. A model that learns that "Paris" and "France" are semantically related has learned something about the world, not merely about syntactic co-occurrence. But the learning is mediated entirely by linguistic context, without any grounding in the sensorimotor experience of the entities named. The model's representation of "Paris" encodes its distributional relations to other linguistic items, not any experiential contact with the city.

The Spherepop framework approaches grounding differ-

ently. Grounding, on the admissibility-manifold view, is not a matter of connecting symbols to sensorimotor experience through a separate grounding relation. It is a matter of the admissibility conditions being determined by the actual structure of the domain being represented. A grounded representation of arithmetic is one whose admissibility conditions reflect the actual constraint structure of arithmetic — the nesting relations that determine evaluation order, the binding conditions that determine scope, the collapse conditions that determine when reduction is admissible. A representation that correctly encodes these admissibility conditions is grounded in arithmetic in the relevant sense, regardless of whether it is accompanied by sensorimotor experience of numbers. Grounding is constraint fidelity: the admissibility manifold of the representation faithfully reflects the admissibility manifold of the domain.

Chapter 20

Ontological Structure and Grounded Representation

Meaning is analyzed as relational constraint: semantic neighborhoods, language as structured navigation, and grounded representation are developed as the ontological foundations for a Spherepop-based semantics.

20.1 Meaning as Relational Constraint

The view of meaning developed in this monograph is resolutely relational. Meaning is not a property of individual expressions considered in isolation — not a mental image, not a neural activation pattern, not a pointer to an object in the world. It is a structural property of the position an expression occupies within a network of admissibility relations: the set of constraints that govern its combination with other expressions, the set of transitions it licenses and the set it forecloses, the set of contexts in which its use is appropriate and the set in

which it is inadmissible. Two expressions have the same meaning, in this sense, if and only if they occupy the same position in the admissibility network — if they license the same transitions, foreclose the same alternatives, and are contextually appropriate in exactly the same set of circumstances.

This view of meaning has deep roots in the philosophical tradition. It is related to Wittgenstein's use-theoretic conception of meaning — meaning as governed by the rules of a language-game — and to the inferentialist semantics developed by Brandom, on which the meaning of an expression is constituted by the inferential role it plays in the practices of giving and asking for reasons. It is also related to the distributional hypothesis in computational linguistics — the idea that words with similar meanings appear in similar contexts — though the Spherepop formulation is more precise: the relevant similarity is not distributional co-occurrence but structural identity of the admissibility conditions governing use.

The relational conception of meaning is not merely a philosophical preference. It has formal consequences that distinguish it from both atomistic and purely inferentialist alternatives. Because meaning is constituted by admissibility relations, and because admissibility relations are formally characterized by the constraint structures of the Spherepop framework, meaning is subject to the same formal analysis as admissibility more generally: it has a geometry (the admissibility manifold), a topology (the structure of semantic neighborhoods), a dynamics (the process of semantic interpretation as trajectory selection through the manifold), and a thermodynamics (semantic entropy as the volume of the admissibility set). These are not metaphors imposed on meaning from outside; they are formal

consequences of treating meaning as a structural property of relational position within a constraint network.

20.2 Semantic Neighborhoods

The notion of a semantic neighborhood formalizes the intuition that meanings are not isolated points in a conceptual space but structured regions: clusters of closely related expressions that share large portions of their admissibility conditions and differ in others, forming a continuous topology over the space of meanings. Two expressions are semantically close — they occupy the same neighborhood — if their admissibility conditions largely coincide: the contexts in which one is appropriate are largely the contexts in which the other is appropriate, the inferences one licenses are largely the inferences the other licenses, and the constraints one imposes are largely the constraints the other imposes.

Definition 20.1. The *semantic neighborhood* $\mathcal{N}_\epsilon(e)$ of an expression e at radius ϵ is the set of expressions e' such that the symmetric difference of their admissibility conditions has measure at most ϵ:

$$\mathcal{N}_\epsilon(e) = \{e' : |\mathsf{Adm}(C_e) \mathbin{\triangle} \mathsf{Adm}(C_{e'})| \leq \epsilon\},$$

where C_e and $C_{e'}$ are the constraint configurations governing the use of e and e' respectively.

Semantic neighborhoods are not fixed but context-dependent: the admissibility conditions governing an expression's use vary with context, and therefore so do the boundaries of the neighborhood. In a mathematical context, "function" and "mapping"

may occupy the same neighborhood, while in an organizational context they diverge. In a philosophical context, "concept" and "idea" may be nearly synonymous, while in a psychological context they are sharply distinguished. The context-dependence of semantic neighborhoods is formally captured by the context-dependence of the constraint configuration C_e: different contexts impose different constraints, and the neighborhood is determined by the constraints in force.

20.3 Language as Structured Navigation

If meaning is relational and constituted by admissibility conditions, then the use of language is navigation through a structured space of meanings: the selection, from a position in the admissibility manifold, of a next expression that is contextually admissible given the current constraint configuration. Producing a sentence is traversing a trajectory through semantic space, with each word choice constrained by the words that preceded it and by the extra-linguistic context. Understanding a sentence is reconstructing the trajectory from its surface form, recovering the sequence of semantic positions traversed and the constraint conditions that governed each step.

This navigational picture of language use is not merely a metaphor. It has precise formal content within the Spherepop framework. A linguistic trajectory is a sequence of semantic positions $s_0, s_1, \ldots, s_n$ through the admissibility manifold, connected by transitions that are locally admissible given the constraint configuration at each step. The constraint configuration is determined jointly by the grammar of the language, the prior linguistic context, the extra-linguistic situation, and the shared

background knowledge of speaker and hearer. Grammaticality is local admissibility: a string is grammatical if each transition in the trajectory it traces is locally admissible given the grammatical constraints in force. Semantic appropriateness is global admissibility: the full trajectory is semantically appropriate if the total action cost — the sum of contextual mismatches and constraint violations across all steps — is below the threshold of communicative success.

The navigational picture explains several features of language use that are puzzling on atomistic accounts of meaning. Ambiguity is high semantic entropy at a given point in the trajectory: many different next positions are locally admissible, and the constraint configuration does not uniquely determine which to take. Disambiguation is entropy reduction: additional context constrains the admissibility set, ruling out alternatives and selecting a unique trajectory. Metaphor is locally inadmissible transition that is globally coherent: the literal meaning of the expression is inadmissible in the current context, but a nearby semantic position — reachable by a small perturbation of the admissibility conditions — is admissible and communicatively productive.

20.4 Grounded Representation

The account of grounding as constraint fidelity is elaborated here into a positive theory of grounded representation. A representation is grounded in a domain D to the extent that its admissibility conditions faithfully reflect the admissibility conditions of D itself: the transitions the representation permits are the transitions that D permits, and the transitions it forecloses

are the transitions that D forecloses. A perfectly grounded representation of D is one whose admissibility manifold is isomorphic to the admissibility manifold of D — a structure-preserving map between the two constraint networks that takes admissible transitions in the representation to admissible transitions in D and vice versa.

This is a demanding standard, and most real representations fall short of it. A textbook's representation of arithmetic permits some transitions that arithmetic forbids (students make errors that the notation does not immediately flag) and forecloses some transitions that arithmetic permits (students who learn arithmetic through rote memorization of procedures may be unable to perform legitimate transformations not covered by the memorized rules). The textbook's admissibility manifold is an imperfect approximation to arithmetic's admissibility manifold, with both false positives and false negatives.

Spherepop notation, in the domain of arithmetic, is an attempt to minimize false negatives: by making the admissibility conditions explicit in the visual structure of the bubble topology, the notation ensures that every admissible transition is visually legible and every inadmissible transition is visually blocked. The evaluation order of $1 + 3 \times 2^2$ is not merely inferred from memorized rules; it is directly visible in the nesting structure of the bubbles, which accurately reflects the constraint structure of arithmetic's evaluation order. The representation's admissibility manifold is better aligned with arithmetic's admissibility manifold, and grounding is thereby enhanced.

Chapter 21

Spherepop and Explainable Semantics

This chapter develops the positive proposal: Spherepop as a framework for visible semantic operations, operational provenance, and localized meaning regions. Computation is reconceived as navigable geometry, and the chapter provides a treatment of topology, coordinate charts, and sheaf-like structures as the mathematical apparatus of this conception.

21.1 Visible Semantic Operations

The positive proposal of this chapter is the development of Spherepop as a framework for visible semantic operations: a representational system in which semantic transformations are directly readable from the representation, rather than encoded in invisible computational processes or recoverable only through post-hoc explanation. The contrast with conventional symbolic and statistical systems is precise. In conventional symbolic systems, semantic operations are visible in principle

but often presented in compressed form that makes individual steps hard to track and their semantic significance hard to read. In statistical systems, semantic operations are invisible in practice: the learned parameters that implement the operations are distributed across millions of weights with no individual weight carrying interpretable semantic content.

Spherepop makes semantic operations visible at three levels. At the notational level, the bubble topology directly encodes scope, dependency, and evaluation order: the semantic structure of the expression is readable from its visual structure without inference or reconstruction. At the operational level, each pop event is a visible semantic transformation: it is the explicit resolution of a local semantic question — what is the value of this subexpression? — whose answer is recorded in the evaluation history. At the provenance level, the full sequence of pop events is a visible record of the semantic trajectory: the history of how the current semantic state was reached from the initial expression, which alternatives were foreclosed at each step, and what the admissibility conditions were that governed each transition.

21.2 Operational Provenance

Provenance, in the context of Spherepop, is the full record of the causal history of a semantic or computational result: the ordered sequence of evaluation steps by which the result was reached, together with the constraint configurations in force at each step, the alternatives that were locally admissible but not taken, and the commitment costs incurred along the way. Provenance is not merely a debugging tool or an auditing mech-

anism. It is, within the Spherepop framework, a constitutive component of semantic content: the result without its provenance is a terminal state without a trajectory, an answer without an understanding of the question.

The claim that provenance is constitutive rather than auxiliary is the Spherepop framework's strongest departure from conventional computational ontology. In conventional computation, a program's output is its semantic content: two programs that produce the same output on all inputs are extensionally equivalent, and their operational differences are implementation details with no semantic significance. In Spherepop, two programs that produce the same output via different histories are semantically distinct under any collapse operator that preserves provenance: they occupy different positions in the admissibility manifold, they have different remaining-freedom profiles, and they support different inferences about the structure of the computation that produced the result.

Proposition 21.1. *Let γ_1 and γ_2 be two Spherepop histories with the same terminal value but different admissibility profiles. Under any collapse operator q that is admissibility-preserving, $\gamma_1 \not\sim_q \gamma_2$. That is, provenance-preserving collapse distinguishes histories that terminal-value-only collapse identifies.*

21.3 Localized Meaning Regions

The Spherepop framework treats meaning as *local*: the semantic content of a subexpression is constituted by the constraint conditions governing its evaluation within its scope region, and that content is not determined by global properties of the full expression in which it is embedded. This locality principle is

one of the framework's most distinctive features and one of its most important practical commitments.

Locality of meaning has both a geometric and a cognitive justification. Geometrically, locality corresponds to the fact that the admissibility conditions governing a bubble's contents are determined by the bubble's boundary conditions — the constraints imposed by the containing bubbles at the boundary of the scope region — and not by the details of what lies outside those boundaries. A bubble is a semantically autonomous region: its internal evaluation is governed by local admissibility conditions that are, in principle, understandable without reference to the full context. Cognitively, locality corresponds to the chunking mechanisms described in Part II: meaningful units are cognitively processed as wholes, and the cognitive resources required to understand a unit are proportional to the complexity of the unit rather than to the complexity of the larger structure in which it is embedded.

21.4 Computation as Navigable Geometry

The reframing of computation as navigation through semantic configuration space is consolidated here as a philosophical thesis about the nature of computation itself. Computation is not, at the most fundamental level, the execution of a set of instructions by a processor according to a fixed algorithm. It is the navigation of a space of admissible transformations, where the space is structured by the constraint conditions in force, and where the navigator moves through the space by selecting locally admissible transitions that collectively constitute a trajectory to the desired terminal state.

The navigational conception differs from the executionist conception in several respects that are philosophically significant. In the executionist conception, computation is deterministic, sequential, and history-indifferent: given the same input, the computation follows the same steps in the same order, and the history of execution is irrelevant to the semantic content of the output. In the navigational conception, computation is structured, trajectory-sensitive, and history-constituted: different trajectories through the same admissibility manifold may reach the same terminal state, but they are semantically distinct; the full trajectory is a primary semantic object, and the terminal state is a compressed summary of the trajectory. The executionist conception optimizes for efficiency and correctness of terminal output. The navigational conception optimizes for the preservation of the structural information encoded in the trajectory.

21.5 Coordinate Charts and Context Windows

The mathematical apparatus of differential geometry provides a natural language for several of the Spherepop framework's central concepts. A *coordinate chart* is a local homeomorphism from a region of a topological manifold to a region of Euclidean space: it assigns coordinates to the points of the region, making local computations tractable, without requiring the global structure of the manifold to be flat or Euclidean.

In the Spherepop framework, a context window — the local context in which an expression is being evaluated, including the constraint configuration, the evaluation history up to the current point, and the admissibility conditions in force — is

a coordinate chart over the admissibility manifold of the full computation. It assigns local semantic coordinates to the current state of the computation, making the local admissibility conditions explicit and the local evaluation decisions tractable, without requiring the global structure of the admissibility manifold to be fully specified. Different context windows cover overlapping regions of the admissibility manifold, with transition functions describing how the local semantic coordinates transform under evaluation.

The coordinate-chart picture gives precise content to several intuitions about context-dependence and semantic continuity. An expression is locally intelligible if it has a well-defined coordinate chart: if the constraint configuration governing its evaluation in the current context is fully specified. It is globally intelligible if the local charts covering all relevant contexts are consistent: if the transition functions between context windows are well-defined and compose correctly. Semantic ambiguity is a failure of local determinacy: the local chart does not uniquely determine the admissible transitions from the current state. Semantic incoherence is a failure of global consistency: the local charts covering different aspects of the expression's use fail to compose into a consistent global picture.

21.6 Sheaf-Like Structures

The mathematical notion of a *sheaf* generalizes the coordinate-chart picture to arbitrary topological spaces. A sheaf over a topological space X assigns to each open set $U \subseteq X$ a set $\mathcal{F}(U)$ of local data — functions, sections, semantic content — together with restriction maps that describe how local data on a larger

open set restricts to local data on a smaller one, satisfying locality (a section is determined by its restrictions to a covering) and gluing (compatible local sections can be assembled into a global section).

In the Spherepop framework, the sheaf-theoretic picture formalizes the local-to-global assembly of semantic content. The admissibility manifold $\mathcal{A}(X)$ of a computation is the base space. The sheaf assigns to each open region U of the admissibility manifold the set of locally admissible evaluation sequences over U — the set of trajectories that are consistent with the local constraint configuration within U. The gluing axiom says that locally admissible trajectories that are consistent on overlapping regions can be assembled into a globally admissible trajectory — precisely the local-to-global assembly that supports the construction of fully grounded meanings from locally grounded components. The failure of sheaf-like assembly — the condition in which locally consistent context windows fail to produce a globally consistent meaning — corresponds precisely to semantic incoherence.

21.7 Local-to-Global Meaning Construction

The local-to-global assembly of meaning is the process by which individually intelligible components are composed into a globally coherent semantic object. In Spherepop, it is trajectory assembly: the globally admissible evaluation trajectory is constructed from locally admissible evaluation steps, with each step's local admissibility ensuring that it can be coherently incorporated into the global trajectory.

The local-to-global process is genuinely compositional in

the sense that the meaning of the whole is not merely the sum of the meanings of the parts, but a structured function of those meanings mediated by the combinatorial rules — the bubble topology, the admissibility conditions at each boundary — that govern their assembly. A bubble containing 3×2^2 has a local meaning determined by the local admissibility conditions within the bubble: the inner bubble nesting that encodes the rule that exponentiation is evaluated before multiplication. When this bubble is incorporated into the larger expression $1 + 3 \times 2^2$, its local meaning is assembled with the local meaning of the outer addition operation according to the admissibility conditions at the bubble's boundary, producing a globally coherent semantic object whose meaning is the structured composition of its parts' meanings.

21.8 Semantic Continuity

Semantic continuity is the property of a representation that guarantees that small changes in the represented domain produce small changes in the representation's admissibility conditions. A semantically continuous representation supports reliable inference: if you know the semantic content of nearby expressions, you can predict the semantic content of the expression in question with small error.

Semantic continuity is an important structural property because its failure — semantic discontinuity — is one of the primary sources of interpretability failure in AI systems. A system whose output changes dramatically in response to small, semantically irrelevant changes in input is exhibiting semantic discontinuity: the map from input semantic content to output

is not continuous in the relevant topology. This is the phenomenon of adversarial examples in neural networks: small perturbations of the input that are semantically negligible to human observers produce dramatic changes in the network's output, revealing that the network's implicit semantic space has a topology that diverges sharply from the topology of the intended semantic domain.

Within the Spherepop framework, semantic continuity is guaranteed by the locality principle and the sheaf-like structure of meaning assembly. Because each bubble's local admissibility conditions are determined by the constraints at its boundary, and because small changes in the expression outside a bubble leave those boundary constraints unchanged, the local meaning of the bubble is semantically continuous with respect to changes in the outer expression.

Part VII

Geometry, Waves, and Physical Representation

Chapter 22

Wave Structure and Relational Physics

Standing waves, interaction, collapse and revival, and localized coherence are examined as physical instances of the structures Spherepop formalizes computationally. Phase space and constraint fields are developed as the physical counterparts of Spherepop's semantic regions.

22.1 Standing Waves and Interaction

The physical picture of a standing wave offers one of the most productive structural analogies available for the Spherepop framework's treatment of localized, bounded, self-sustaining semantic regions. A standing wave is a pattern of constructive and destructive interference between waves traveling in opposite directions that produces a spatially fixed amplitude envelope: a pattern of nodes, where amplitude is permanently zero, and antinodes, where amplitude is permanently maximal. The pattern is sustained not by any static mechanism but by the ongoing dynamical interaction of the traveling waves that

compose it. It is a structure that exists in time as a stable product of dynamical process, not as a static object independent of its generative dynamics.

The analogy to Spherepop bubbles is structural. A bubble is not a static container but a dynamically sustained evaluative region: its boundaries are maintained by the ongoing constraint conditions at the bubble's interface, and its internal evaluation is sustained by the local admissibility conditions in force within it. Just as a standing wave's pattern is determined by the boundary conditions at its endpoints, a bubble's evaluative behavior is determined by the constraint conditions at its boundary. And just as a standing wave collapses when its boundary conditions change, a bubble pops when its internal evaluation reaches the point at which the local admissibility conditions are fully satisfied and commitment to a result becomes possible.

The deepest point of the analogy is the relationship between local structure and global coherence. A standing wave is a globally coherent pattern produced by locally determined phase relations: the interference is constructive at antinodes because the phases of the component waves align there, and destructive at nodes because the phases cancel. Similarly, a globally coherent Spherepop computation is produced by locally admissible evaluation steps: the global trajectory is coherent not because it was determined from outside but because each local step satisfied the constraint conditions in force at that point.

22.2 Collapse and Revival

The phenomenon of collapse and revival in quantum mechanical systems provides a physical instance of a structural pattern

that is central to the Spherepop framework: the alternation between coherent, locally organized states and collapsed, globally disordered ones, followed by spontaneous re-emergence of coherence. In a collapsing quantum state, initially coherent superpositions of energy eigenstates dephase over time as the different components accumulate different phases, producing apparent incoherence and loss of the initial quantum structure. Revival occurs when the phases realign sufficiently to regenerate the initial coherent pattern, demonstrating that the structure was not destroyed but merely hidden in the dephased superposition.

The structural parallel with Spherepop collapse is exact in the relevant sense: collapse in both cases is not destruction but transformation into a less accessible form. A Spherepop collapse event $\mathsf{collapse}(q)$ maps a history γ to an equivalence class $\gamma/\sim_q$, identifying all histories that are observationally equivalent under the chosen collapse operator. The structural information that distinguishes γ from other histories in its equivalence class is not destroyed; it is encoded in the kernel of the quotient map. Revival — the recovery of that structural information from the collapsed description — is formally possible whenever the kernel is recoverable from the quotient map. In practice, revival may be impossible if the kernel information has been discarded rather than merely suppressed, exactly as quantum revival is impossible if the initial state information is truly lost through irreversible decoherence rather than merely obscured through dephasing.

22.3 Localized Coherence

Coherence, in the physical sense, is the maintenance of definite phase relations among the components of a superposition over time. A coherent state is one in which the phases are sufficiently aligned to produce constructive interference in observable quantities; an incoherent state is one in which the phases are randomized, suppressing interference effects.

The Spherepop analogue of coherence is the maintenance of consistent admissibility conditions across the components of a computation — the condition in which the constraint configurations governing different parts of the evaluation are mutually consistent, and in which the local admissibility conditions at each bubble boundary compose correctly into a globally admissible evaluation trajectory. A *semantically coherent* computation is one in which all active bubbles have consistent constraint configurations, all admissibility conditions are simultaneously satisfiable, and the evaluation trajectory is well-defined at each step. A *semantically incoherent* computation is one in which constraint configurations conflict, admissibility conditions cannot all be simultaneously satisfied, or the evaluation trajectory is blocked by mutually incompatible constraints.

Localized coherence — coherence maintained within a bounded region while the surrounding environment is incoherent — corresponds in the Spherepop framework to the local admissibility of a bubble's internal evaluation even when the containing context is globally underdetermined. This is the structural property that makes local-before-global evaluation order correct: inner bubbles can be evaluated coherently before the outer context is resolved, because their admissibility conditions are locally determined.

22.4 Phase Space and Constraint Fields

The admissibility manifold of a Spherepop computation is the computational analogue of a constrained mechanical phase space: the space of all admissible computation states, with the constraint conditions — the bubble topology, the active bind constraints, the refuse events — defining the admissible region. Just as a mechanical constraint reduces the available phase space from the full unconstrained manifold to the constraint surface, a Spherepop constraint configuration reduces the available computation space from the full set of all evaluation sequences to the admissibility manifold $\mathcal{A}(X)$.

The constraint field over the admissibility manifold — the function that assigns to each point in the manifold a specification of the constraint configuration in force — is the Spherepop analogue of the Hamiltonian vector field in mechanics: it determines the direction of evolution at each point. Where the mechanical Hamiltonian generates infinitesimal displacements along the constraint surface, the Spherepop constraint field specifies which evaluation steps are locally admissible at each point of the admissibility manifold. The geometry of the constraint field — its singularities, attractors, and repellers — determines the global structure of the evaluation dynamics: which terminal states are accessible from which initial states, which trajectories are geodesics of the action functional, and which regions of the admissibility manifold are inaccessible from the current state.

Chapter 23

Blender Models and Computational Topology

Three-dimensional geometric models—orbital structures as semantic regions, spheres as cognitive manifolds, rings as constraint boundaries—are developed as operational languages for visualizing nested computation. The chapter argues that geometry is not merely illustrative but constitutive of the semantics it expresses.

23.1 Orbital Structures and Semantic Regions

The orbital structures developed in three-dimensional geometric modeling — nested spherical shells with intersecting rings, visualized as admissibility boundaries and evaluation trajectories — are not decorative abstractions. They are operational languages for spatial reasoning about nested computation, and their utility is grounded in the cognitive geometry arguments of Part II: human spatial cognition handles nested, bounded regions with natural competence, and visual representations that

exploit this competence support reasoning that flat symbolic representations hinder.

An orbital structure in this context consists of a central region — the innermost bubble, whose evaluation is most locally constrained and whose pop event is the first admissible transition — surrounded by progressively larger shells representing the containing bubbles, each with its own constraint boundary. The rings intersecting the shells represent the bind constraints that connect the evaluation of inner bubbles to the admissibility conditions of outer ones: they are the visible manifestation of semantic dependency, showing which evaluations in outer shells depend on the results of evaluations in inner ones. The evaluation of such a structure proceeds geometrically: each pop is a geometric collapse of a shell to its resolved value, with the constraint rings connecting it to outer shells dissolving as the constraints are satisfied.

23.2 Spheres as Cognitive Manifolds

The choice of spheres as the primary geometric objects in the Spherepop visual representation is not arbitrary. Spheres possess geometric properties that align with the cognitive and formal requirements of the framework. They are maximally symmetric — having no preferred direction, no distinguished boundary points — which reflects the local uniformity of admissibility conditions within a scope region: within a bubble, every point is equivalently subject to the same constraint configuration. They have a well-defined interior and exterior separated by a smooth boundary, which reflects the sharp binary character of local admissibility: a step is either inside the admissibility

set (within the bubble) or outside it, with no graded interior.

Cognitively, spheres are among the most primitive geometric objects in human visual processing. The perception of a closed, smooth boundary enclosing a distinct interior is among the earliest emerging and most robust visual capacities. The sphere's perceptual salience as a unified, bounded object is therefore a genuine cognitive asset: it ensures that the bubble boundary is perceived as a semantic boundary rather than as a decorative element. Formally, a sphere S^2 embedded in $\mathbb{R}^3$ is a topological manifold with a canonical orientation (inside versus outside) and a canonical boundary (the sphere itself). These formal properties correspond exactly to the formal properties of a Spherepop bubble: it has a canonical inside (the scope region), a canonical outside (the containing scope), and a canonical boundary (the bubble's interface).

23.3 Rings as Constraint Boundaries

If spheres represent scope regions, rings — closed curves wrapping around or intersecting spherical shells — represent constraint boundaries: the visible manifestation of the bind constraints that connect evaluations across scope levels. A ring encircling a sphere at a given latitude represents a bind constraint that restricts the evaluation within the sphere's interior to the region satisfying the constraint condition encoded in the ring's position. A ring that intersects two concentric spheres at their shared boundary represents a dependency relation: the evaluation of the inner sphere is constrained by the result it will eventually propagate to the outer sphere.

The visual grammar of rings and spheres, taken together,

forms a spatial notation for the full constraint structure of a Spherepop computation. Concentric spheres represent nested scopes. Rings encircling a sphere represent active bind constraints within that scope. Rings crossing sphere boundaries represent inter-scope dependencies. Gaps in rings represent refuse events. This visual grammar supports spatial reasoning that linear notation does not: a reader can see the constraint density of a region, the depth of nesting, and the inter-scope dependency structure from the diagram's geometry without requiring any textual annotation.

23.4 Visualizing Nested Computation

The full visual representation of a Spherepop computation is a three-dimensional, dynamically evolving structure: a sequence of configurations of nested spheres with rings, in which each pop event is represented as the collapse of the innermost sphere to a point, the dissolution of the rings that connected it to outer spheres, and the update of the outer spheres' configurations to reflect the result of the collapsed evaluation. The full trajectory of the computation — the ordered sequence of pop events — is a sequence of geometric transformations of this three-dimensional structure, each transformation reducing its complexity and advancing the evaluation toward the terminal state.

This dynamic geometric representation is a visual calculus of constraint: a system for performing computations by manipulating visual structures in accordance with geometrically expressed rules. Pop events are collapses of spheres. Bind events are additions of rings. Refuse events are markings on

potential ring positions. The evaluation order is determined by the nesting structure: innermost spheres collapse before outer ones, exactly as the geodesic of the semantic configuration space dictates. The value of this representation is the alignment of the computational structure with the cognitive architecture of spatial reasoning: the working memory cost of tracking the evaluation state is externalized into the three-dimensional structure, making scope boundaries, constraint relations, evaluation order, and evaluation history all simultaneously visible.

Chapter 24

Geometry as Operational Language

Topological semantics, computation as deformation, spatialized logic, and meaning as curvature in constraint space are developed into a unified geometric language for Spherepop. This chapter represents the apex of the physical-geometric thread running through the monograph.

24.1 Topological Semantics

The topological approach to semantics treats the space of meanings not as a discrete set of atomic elements but as a continuous structured space with a topology: a specification of which sets of meanings are "open" — semantically coherent, locally uniform, accessible to the same inferential operations — and which are "closed" — stable under the relevant convergence operations. In the Spherepop framework, the topology on the semantic space is induced by the admissibility conditions: two meanings are topologically close if they have similar admissibil-

ity conditions, and a semantic map is continuous if it preserves the structure of admissibility neighborhoods.

The topological approach gives a precise account of semantic generalization: the extension of a meaning from a domain in which it is well-defined to a domain in which it is not yet specified. Generalization is topological extension: the semantic map is defined on a dense subset of the semantic space and extended to the full space by continuity, taking the limit of the map's values on the known domain as the argument approaches the unknown point. Semantic generalization fails when the topological extension is not the intended one, either because the topology is incorrectly specified or because the map is not genuinely continuous on the known domain. This is a precise formal account of the difference between genuine understanding — which supports correct generalization — and memorization, which does not.

24.2 Computation as Deformation

The geometric picture of computation as deformation — the view that evaluating an expression is a topological deformation of its semantic structure rather than a syntactic transformation of its symbolic form — gives concrete content to the navigational picture of computation. Each pop event is an elementary deformation: the collapse of an innermost sphere to a point, which is a continuous retraction of the sphere's interior onto its center. The retraction is homotopically trivial — it does not change the topological type of the surrounding structure, because the sphere is contractible — but it changes the constraint configuration.

The full sequence of pop events is a composition of elementary deformations, and the resulting total deformation is the homotopy of the admissibility manifold corresponding to the evaluation trajectory. The deformation picture makes visible a structural property of evaluation that is invisible in the executionist framework: the fact that evaluation preserves topological structure while reducing complexity. Each pop event reduces the number of active bubbles while maintaining the topological coherence of the remaining structure. This is compression in the topologically correct sense: structure-preserving reduction of complexity through homotopy, not destruction of structure through erasure.

24.3 Spatialized Logic

The spatialization of logic — the interpretation of logical operations in terms of spatial relations and geometric transformations — has a long philosophical history, from the geometric diagrams of Euler and Venn to the topological interpretations of intuitionistic logic in the internal language of toposes. The Spherepop framework contributes to this tradition by providing a computational realization of spatialized logic in which the logical operations of conjunction, disjunction, negation, and implication are expressed as spatial operations on bubble structures.

Conjunction of two constraints corresponds to their intersection in the admissibility space: a state is admissible under the conjunction if and only if it is admissible under both. This is represented geometrically as the intersection of two constraint regions. Disjunction corresponds to the union of constraint

regions. Negation corresponds to complementation: the admissibility boundary of the negation is the complement of the original constraint region in the admissibility manifold. Implication corresponds to inclusion: $A \Rightarrow B$ is satisfied at a state if that state lies outside the admissibility region of A or inside the admissibility region of B.

The spatial representation of logical operations has a cognitive advantage: it makes logical validity directly readable from the geometric structure rather than inferrable from formal rules. A logically valid inference — one that is truth-preserving — corresponds to a spatial operation that preserves containment in the admissibility region. A logically invalid inference corresponds to a spatial operation that violates this containment. The validity of the inference is directly visible in the containment relationship between the spatial regions.

24.4 Meaning as Curvature in Constraint Space

The most ambitious claim of the geometric program is the interpretation of semantic content as curvature in the admissibility manifold: the view that the meaning of an expression is encoded not merely in its position in semantic space but in the local geometry of the admissibility manifold around that position — in the curvature, the geodesic structure, and the local topology of the manifold in the expression's semantic neighborhood.

Curvature in a Riemannian manifold measures the extent to which parallel transport around a closed curve produces a rotation of the transported vector. In the semantic configuration space, curvature corresponds to the extent to which the admis-

sibility conditions governing an expression's use change as the context varies around the expression: high curvature means that small changes in context produce large changes in admissibility conditions, corresponding to high context-sensitivity and semantic instability. Low curvature means that admissibility conditions are approximately constant across nearby contexts, corresponding to semantic stability and context-independence.

The curvature interpretation provides a geometric account of semantic phenomena. Polysemy — the possession by a single word of several distinct but related meanings — corresponds to regions of high curvature: the admissibility conditions change rapidly as the context shifts, selecting different meanings in different contextual directions. Semantic rigidity corresponds to low curvature: the admissibility conditions are approximately constant across a wide range of contexts. Semantic creativity — the productive extension of an expression's meaning to new contexts through metaphor — corresponds to the smooth extension of geodesics beyond their original domain, following the curvature of the semantic manifold into previously unexplored regions.

Part VIII

Toward a Unified Philosophy of Computation

Chapter 25

From Strings to Spaces

This chapter provides a historical account of the linearization of computation and argues for the recovery of spatial structure. The return of geometry to computation is framed not as a regression to pre-formal intuition but as a dialectical advance: the recovery of structure after its near-total submersion in symbolic formalism.

25.1 The Historical Dominance of Textual Formalism

The history of formal symbolic systems is, to a remarkable degree, the history of a progressive linearization: the reduction of what were originally spatial, diagrammatic, and multi-dimensional representational practices to one-dimensional sequences of symbols arranged on lines of text. The geometry of Greek mathematical diagrams, the spatial layout of medieval logical trees, the two-dimensional notation of early algebraic manuscripts — all of these were gradually displaced, over the course of the early modern period, by the linear symbolic nota-

tion that is now so universal that it appears natural rather than historically contingent.

The linearization was not driven primarily by cognitive considerations. It was driven by the constraints of textual transmission: books, manuscripts, and later printing presses imposed one-dimensional ordering on their content, and mathematical notation evolved to satisfy this constraint. Symbolic algebra, as developed by Viète, Descartes, and their successors, was specifically designed to be printable on a single line of text, with operations encoded as sequences of symbols according to conventions that suppressed the two-dimensional structure visible in earlier notation. The resulting linear notation was extraordinarily productive — it enabled algebraic manipulation at a scale and speed previously impossible — but it achieved this productivity by encoding structural information that had previously been spatial into invisible conventions: operator precedence, implicit parenthesization, associativity, and the like.

25.2 Why Computation Became Linearized

The practical reasons for the linearization of computation are easily identified: linear text is sequential and therefore compatible with sequential execution; it is compact and therefore efficient to store and transmit; and it is the format in which human beings had extensive prior training from natural language reading and mathematical notation. These are genuine advantages, and they explain why linear text proved so productive as a computational medium.

But the advantages are relative to a set of constraints — con-

straints of storage medium, transmission bandwidth, and parsing technology — that were characteristic of the mid-twentieth century and have been substantially relaxed since. Modern computational environments support rich two-dimensional and three-dimensional visual representations, interactive manipulation of structured objects, and parsing technologies capable of handling non-linear input formats. The practical case for linearization has weakened substantially. What remains is cognitive inertia: the investment of a generation of programmers in linear-text conventions, the standardization of linear-text formats in infrastructure, and the difficulty of reconceiving a practice so deeply embedded in professional identity.

The philosophical case for linearization was always weaker than the practical one. Linear text compresses structural information into invisible conventions, and those conventions impose cognitive costs proportional to the complexity of the structural information being suppressed. For complex programs with deep nesting, multi-level scope, and intricate dependency structures, the costs are substantial. And for novel or ambiguous programs whose structure is not yet clear to the programmer, the cost of maintaining an implicit structural model in working memory is precisely the cost that prevents understanding from forming.

25.3 Recovering Spatial Structure

The recovery of spatial structure in computational notation is not a regression to pre-formal intuition. It is the recognition that spatial structure is a genuine carrier of semantic content — that scope, dependency, evaluation order, and admissibility

are spatial properties of the computation that deserve spatial representation — and that the suppression of this structure in linear notation was a practical accommodation to medium constraints rather than a principled semantic choice.

Recovering spatial structure means allowing the visual organization of a computational representation to carry semantic information directly: scope is containment, not convention; evaluation order is nesting depth, not precedence rule; dependency is proximity and connection, not invisible reference; admissibility is boundary, not implicit context. The recovery of spatial structure is already underway in many areas of computational practice. Type-theoretic proof assistants display proofs as trees. Dataflow programming environments display programs as graphs of data transformations. Visual programming languages for education use spatial arrangement and containment to communicate scope and control flow. Each of these is a partial realization of the Spherepop principle: let the visual structure of the representation carry the semantic structure of the computation.

25.4 The Return of Geometry

The return of geometry to computation is the recognition that the fundamental objects of computation — histories, admissibility manifolds, constraint configurations, evaluation trajectories — are geometric objects, and that their natural representation is geometric rather than linear. The history of a computation is a path through a manifold. The admissibility conditions are a constraint field over that manifold. The evaluation order is the geodesic of the action functional. The terminal value is the

endpoint of the geodesic. All of these are geometric objects, and all of them are more directly expressed in geometric notation than in linear symbolic notation.

The return of geometry is also the recognition that human cognition is, at its most fundamental level, a geometric activity: the navigation of a structured spatial environment using spatial reasoning capacities that are evolutionarily ancient, neurologically robust, and cognitively primary. Symbolic reasoning is a late evolutionary addition, culturally mediated, individually learned, and cognitively expensive. Spatial reasoning is a biological heritage, culturally universal, individually innate, and cognitively cheap. A representational system that grounds its formal operations in spatial reasoning rather than imposing them through symbolic conventions is not making a concession to cognitive weakness. It is exploiting a cognitive strength.

Chapter 26

Computation as Navigation

Trajectories through admissibility space, constraint satisfaction and future possibility, local actions and global consequences, and histories as fundamental objects are developed into a comprehensive philosophy of computation organized around the concept of navigation rather than execution.

26.1 Trajectories Through Admissibility Space

Computation is the selection and traversal of an admissible trajectory through a structured space of possibilities, where the space is defined by the constraint conditions of the problem, the trajectory is the history of evaluation steps by which the terminal state is reached, and the selection of the trajectory is governed by the action functional that measures its total structural cost. The terminal value — the output of the computation — is the endpoint of the trajectory, and it is meaningful only as such: as the terminal state of a specific history within a specific constraint space, not as a value existing independently of the process that produced it.

This thesis has four components, each developed in previous chapters. The admissibility space is the space of all possible computation states that satisfy the admissibility conditions of the system: a geometric object with a topology induced by the constraint structure and a boundary defined by the binary admissibility conditions. Trajectories are paths through this space: ordered sequences of admissible evaluation steps that move from initial to terminal state. The action functional assigns a cost to each trajectory. Navigation is the process of selecting trajectories through the admissibility space in accordance with the action functional, guided by the Principle of Least Structural Commitment.

26.2 Constraint Satisfaction and Future Possibility

Constraint satisfaction — the condition in which all active constraints are simultaneously satisfied by the current computation state — is the prerequisite for continued evaluation: only locally admissible states support further evaluation steps. Future possibility, measured by the remaining accessibility volume Ω, is the primary dynamic quantity of a computation in progress. A computation with high Ω retains many alternative trajectories to the terminal state; a computation with low Ω is nearly committed to a specific trajectory, with the terminal state almost determined by accumulated constraints.

The decrease of Ω over the course of a computation is the computational analogue of the increase of entropy over the course of a thermodynamic process: it is the irreversible consumption of future possibilities in the act of producing a de-

terminate result. The question for the design of computational systems is not whether irreversibility should occur, but whether it should be visible. A system that makes each reduction of Ω visible — that records each pop event as an explicit commitment with its associated constraint configuration and action cost — is a system in which the irreversibility of evaluation is exposed to scrutiny. A system that performs the same reductions invisibly, presenting only the terminal value, is a system in which the irreversibility is hidden, and the question of which alternatives were foreclosed and why cannot be answered from the system's output alone.

26.3 Local Actions and Global Consequences

In the executionist framework, local evaluation steps are implementation details: the programmer specifies what the computation should produce, and the implementation determines how it is produced. The local steps are invisible to the semantic description of the computation, which is given entirely in terms of the mapping from inputs to outputs.

In the navigational framework, local evaluation steps have global semantic consequences because the admissibility manifold has global topological structure. A local pop event may foreclose evaluation trajectories that would have been accessible from a different local choice, even if both choices produce the same intermediate value. The global trajectory is not merely the aggregate of local choices; it is shaped by the topology of the admissibility manifold, which may make some global trajectories accessible only through specific local sequences and others inaccessible from certain local positions. This is the com-

putational analogue of the observation in differential geometry that the holonomy of a connection — the rotation produced by parallel transport around a closed curve — is a global property of the manifold's curvature that cannot be determined from any local measurement.

26.4 Histories as Fundamental Objects

A Spherepop history $\gamma = (e_1, e_2, \ldots, e_n)$ is a first-class mathematical object in the following precise sense: it is not merely a record of an otherwise-complete computation but a constitutive component of the computation's semantic content. The history determines: the terminal value (through the composition of evaluation steps); the admissibility profile (the sequence of constraint configurations in force at each step); the commitment trajectory (the profile of Ω reduction over the evaluation); and the action cost (the total structural cost of the trajectory). All four are properties of the history, not of the terminal value.

Theorem 26.1. *Let γ_1 and γ_2 be Spherepop histories with the same terminal value but different admissibility profiles. Then there exists an admissibility-preserving collapse operator q such that $\gamma_1 \nsim_q \gamma_2$. Consequently, histories are not determined by their terminal values, and terminal values are not sufficient to determine the semantic content of a computation in the Spherepop framework.*

The theorem formally establishes the inadequacy of the extensional account of computational semantics: the account on which two computations with the same input-output behavior are semantically identical. In Spherepop, extensional identity is a special case of the general contextual equivalence, obtained

by choosing a collapse operator that preserves only terminal values and discards all admissibility information.

Chapter 27

The Ethics of Interpretability

The political and ethical dimensions of computational opacity are confronted directly. Invisible systems, transparency and human cognition, the political dimension of abstraction, and publicly legible computation are argued to be not merely pragmatic concerns but matters of epistemic justice.

27.1 Against Invisible Systems

The argument of this monograph culminates in an ethical claim: that computational systems whose operational processes are invisible to the people they affect are not merely epistemically inadequate but ethically problematic. The ethical claim follows from the epistemic one: if genuine understanding of a process requires access to its history, and if invisible systems do not provide access to their histories, then invisible systems prevent the people affected by them from genuinely understanding what is being done to them and why. This is not a technical failure. It is a failure of the conditions of epistemic agency: the

conditions under which a person can be a genuine participant in the processes that govern their life rather than merely a subject of those processes.

The ethical claim is distinct from the pragmatic argument for transparency that dominates the current discourse on AI governance. The pragmatic argument holds that transparent systems are easier to audit, correct, and govern than opaque ones, and that transparency therefore serves the practical interests of safety and accountability. This is true, but it is not the deepest reason that transparency matters. The deeper reason is that opacity, at the scale at which modern computational systems operate, is a form of structural disempowerment: it removes from the people affected by a system the cognitive resources necessary to understand, contest, or meaningfully consent to what the system is doing.

27.2 Transparency and Human Cognition

The argument for transparency is not merely a claim about information access — the provision of more data — but a claim about cognitive access: the provision of structural information in a form that human cognition can engage with directly. A system that discloses its internal parameters — all several billion of them — has provided complete information access while providing no cognitive access whatsoever. Human cognition cannot navigate billions of undifferentiated numerical parameters to reconstruct the reasoning process that produced a specific output. Complete information disclosure of this kind is, from the perspective of cognitive access, indistinguishable from opacity.

What cognitive access requires is information structured in a form that aligns with the cognitive architecture of the people who need to use it: hierarchically organized, locally bounded, semantically chunked, and geometrically arranged so that the structural relations are visually readable rather than inferentially reconstructed. This is precisely what the Spherepop framework provides. The evaluation history of a Spherepop computation is not a dump of raw parameters but a structured narrative: an ordered sequence of admissible transitions, each locally bounded and semantically meaningful, arranged in a nested hierarchy that makes the dependency structure directly visible.

27.3 The Political Dimension of Abstraction

The political consequences of abstraction — of the systematic replacement of structural complexity by compressed formal summaries — are a recurring theme in the sociology of knowledge and the critique of technocratic governance. Abstraction in the political sense is the reduction of complex social reality to a set of formally tractable indicators: the reduction of health to measurable biomarkers, education to test scores, productivity to tracked metrics, creditworthiness to a numerical score. The indicators may be genuinely informative — they track real features of the phenomena they measure — but they are lossy compressions that discard the structural relations, the historical context, and the local particularity of the situations they summarize.

The political problem with abstraction is not that it is inaccurate but that it is opaque. When a credit score determines

access to housing, the person denied housing cannot determine from the score alone which features of their financial history drove the outcome, which historical contingencies produced those features, or which aspects of their situation the score has failed to capture. The score is a terminal value without a history, a result without a trajectory, a summary without a provenance. And in the absence of provenance, the score cannot be contested on the grounds that its inputs were themselves the product of structurally unjust processes. The Spherepop principle applies directly: the ethical problem with opaque algorithmic systems is that they sever the connection between outcome and history that is the precondition for meaningful accountability.

27.4 Publicly Legible Computation

The concept of publicly legible computation describes the condition under which the computational processes that govern collective life — the algorithms that allocate resources, filter information, assess risk, and distribute opportunity — are expressed in forms that are accessible to the people they affect. Public legibility is not equivalent to public availability of source code. It is the condition in which the computation's operational structure — its admissibility conditions, its evaluation history, its constraint configurations — is expressed in a form that the affected public can genuinely understand and engage with.

Public legibility is a political and democratic value. Democratic accountability requires that citizens be able to assess, contest, and ultimately govern the systems that affect their lives. This requires that those systems be legible: that their operations be expressible in a form that supports genuine pub-

lic understanding, not merely expert technical analysis. The Spherepop framework provides a theoretical model for publicly legible computation: computation in which the evaluation history is part of the system's output, the constraint conditions are visible in the system's representation, and the admissibility of each step is directly readable from the system's structure. The theoretical model demonstrates that there is no in-principle barrier to interpretability. The barrier is practical and political, not theoretical.

Chapter 28

The Future of Structural Thinking

Educational applications, explainable AI, semantic infrastructure, and cognitive geometry as human augmentation are developed as the practical research program entailed by the monograph's theoretical commitments. The chapter closes with a prospectus for a new structural literacy adequate to the computational age.

28.1 Educational Applications

The educational implications of the Spherepop framework follow directly from the analysis of Part II. If scope visualization reduces working memory load, if nested containment is a cognitively primary spatial relation, and if understanding is stabilized navigational compression rather than rote memorization of formal rules, then educational systems designed around these principles will produce qualitatively different outcomes from systems designed around conventional symbolic instruction.

The most immediate application is in mathematics education. The introduction of arithmetic through Spherepop notation would give students direct perceptual access to the structure of arithmetic operations that conventional notation encodes invisibly. A student who learns arithmetic through bubble notation learns not merely which operations to perform but why they must be performed in a specific order: because the nesting structure demands it, because the inner bubble must be resolved before the outer one can be evaluated. This is understanding rather than rote compliance: the student has internalized the operational geometry of arithmetic, not merely memorized its surface rules.

The application extends to programming education, formal logic, proof theory, and natural language grammar — all formal systems in which nested scope and dependency structure are central. Spherepop-style representations that make these relations visually explicit would reduce the working memory cost of learning these systems, increase the rate of genuine understanding over rote memorization, and make the structural principles of the systems accessible to students who currently struggle with purely symbolic notation.

28.2 Explainable AI

The Spherepop framework's contribution to the explainability of AI systems is not primarily a set of techniques for generating post-hoc explanations of black-box models. It is a set of design principles for building systems whose operational processes are interpretable by construction. Post-hoc explanation techniques — saliency maps, attention visualization, decision

tree approximation — are applied to systems whose internal operations are opaque, and the resulting explanations are approximations to the system's local behavior that may or may not faithfully represent its actual computational process.

Spherepop-style design, by contrast, produces interpretability at the level of the system's representation: the system's operational process is expressed in a form that is directly readable, with admissibility conditions explicit, evaluation history preserved, and constraint configurations visible. A system designed on Spherepop principles would expose, for each output it produces, the sequence of locally admissible transitions by which the output was reached, the constraint configurations in force at each step, and the alternatives that were locally available but not taken. This is genuine interpretability, not post-hoc explanation, and it supports the full range of accountability operations that post-hoc explanation does not. The objection that interpretable systems are necessarily less capable than opaque ones is not supported by the theoretical analysis of this monograph. The trade-off between interpretability and capability is not inherent to the problem; it is an artifact of designing for capability first and treating interpretability as an afterthought.

28.3 Semantic Infrastructure

Semantic infrastructure is the underlying computational and representational architecture that makes large-scale semantic systems interpretable, navigable, and accountable. Current semantic infrastructure is largely statistical: it encodes semantic relations in learned numerical parameters, supports navigation through statistical proximity rather than structured ad-

missibility, and provides accountability only through post-hoc explanation of terminal outputs.

The Spherepop framework suggests an alternative architecture in which semantic relations are encoded as admissibility conditions rather than statistical associations, navigation is guided by the topology of the admissibility manifold rather than by numerical proximity, and accountability is provided through the preservation of evaluation history and constraint configuration. Statistical learning can be used to approximate admissibility conditions from data, but the result of the learning is expressed in an admissibility manifold rather than in a numerical parameter vector, making the learned structure interpretable and navigable. The most important property of Spherepop-style semantic infrastructure is its compositionality: because meaning is constituted by admissibility conditions, and because admissibility conditions compose through the sheaf-like assembly described in Chapter 21, the semantic content of a complex expression can be assembled from the semantic contents of its components in a principled, structure-preserving way.

28.4 Cognitive Geometry and Human Augmentation

The concept of cognitive geometry — the use of spatial and geometric representations to support reasoning that would otherwise impose prohibitive working memory costs — points toward a vision of human cognitive augmentation that is distinct from the dominant paradigm of AI-based cognitive extension. The dominant paradigm envisions AI systems that

perform cognitive tasks on behalf of human beings: answering questions, generating text, making decisions, summarizing information. On this paradigm, the AI system is a substitute for human cognition.

The cognitive geometry paradigm envisions a different relationship: not substitution but amplification. A representational system that externalizes the cognitive load of scope tracking, dependency management, and evaluation order — that makes these structural relations directly visible in the geometric structure of the representation — amplifies human cognitive capacity without substituting for it. The human being does the reasoning; the representational system provides a cognitive prosthetic that extends the range and complexity of the reasoning that is cognitively tractable. The reasoning itself remains human, grounded in human understanding, and accountable to human judgment.

This is the vision that Spherepop, taken as a philosophical prototype, points toward: not the replacement of human cognition by opaque computational processes but the augmentation of human cognition by transparent representational structures. The goal is not to make machines that think instead of humans but to make representations that make it easier for humans to think — that externalize the cognitive costs of structural complexity, that make operational geometry visible, and that preserve the provenance and history of reasoning processes in forms that support genuine human understanding.

Conclusion: The Map and the Territory Revisited

The conclusion gathers the monograph's threads. Compression is shown to be necessary and dangerous in equal measure, and the difference between compression that preserves structure and compression that erases it is argued to be the central distinction of our computational moment. Spherepop is presented as a philosophical prototype for a new structural literacy—a visual calculus of constraint, computation as historical geometry, and the recovery of visible meaning.

Why Compression Is Necessary

No intelligent system — biological or computational — can operate without compression. The world is more complex than any representation of it, and intelligence is the capacity to navigate the world by maintaining representations that are simpler than the world itself while remaining faithful to the structural relations that matter for the navigation at hand. A cognitive system that attempted to represent every detail of its environment without compression would be overwhelmed before it could take its first action; a formal system that attempted to

express every fact without abstraction would be unintelligible before it could express its first theorem. Compression is not a failure mode of representation; it is its constitutive operation.

The necessity of compression is not merely practical. It is structural: the very concept of a representation — a structure that stands in for another structure while being simpler than it — is the concept of a compression. To represent is to compress. The question is never whether to compress but how: which structural relations to preserve in the compressed representation and which to discard. Different answers to this question produce different representational systems with different cognitive affordances, different semantic contents, and different epistemic costs.

Why Compression Becomes Dangerous

Compression becomes dangerous when the structural relations that are discarded in the course of compression are precisely the relations needed for genuine understanding of the domain being represented. This is not an abstract possibility; it is a structural tendency of representational systems optimized for a narrow criterion, typically terminal-output accuracy or computational efficiency, at the expense of the richer structural properties that support understanding, contestation, and accountability.

The danger takes three distinct forms. The first is cognitive: when structural relations are compressed into invisible conventions, the cognitive cost of recovering them falls on the user. The second is epistemic: when structural relations are compressed into terminal values, the provenance of those val-

ues is lost, and with it the possibility of genuine understanding of why the result is what it is. The third is political: when structural relations are compressed into opaque algorithmic outputs, the people affected by those outputs lose the cognitive access required for meaningful accountability and democratic governance.

Preserving Structure Through Abstraction

Compression preserves structure when it is an admissibility-preserving collapse: when the equivalence relation under which histories are identified leaves invariant the structural relations that govern the admissibility of further operations. Compression destroys structure when it identifies histories under an equivalence relation that discards admissibility information — when the quotient is determined by the terminal value alone, with no regard for the evaluation order, the constraint configurations, or the commitment costs that were incurred along the way.

The distinction is formal and precise: it is expressed by the factorization condition $f = \bar{f} \circ \mathsf{collapse}(q)$, which says that a compression is admissibility-preserving if and only if every observable function that respects the admissibility structure factorizes uniquely through the compressed representation. The distinction also has a concrete phenomenological content, expressed throughout this monograph in the contrast between the calculator that returns 13 and the Spherepop evaluation that traces the sequence of bubble pops by which the same value is reached. Both achieve compression. But only the second achieves compression that preserves structure: only the second

retains the admissibility information, the evaluation history, and the provenance of the result.

Spherepop as Philosophical Prototype

Spherepop is presented in this monograph not as a finished system but as a philosophical prototype: a proof of concept that demonstrates the possibility of a representational architecture that is simultaneously formally precise, cognitively accessible, semantically transparent, and philosophically coherent. The specific notation, operators, and formal constructions developed here are theoretical tools for articulating and defending the central philosophical claim: that visible structure is possible, that provenance-preserving compression is achievable, and that the trade-off between capability and interpretability is not inherent to computation but is an artifact of design choices that favor opacity.

The value of a philosophical prototype is not its immediate deployability but its existence: the demonstration that the thing is possible in principle. Before Spherepop, the claim that computational representations could preserve evaluation history, make scope visible, and expose admissibility structure while remaining formally precise could be dismissed as a vague aspiration. After Spherepop, the claim has a specific instantiation: a formal system with defined operators, a coherent ontology, an action-functional semantics, and a geometric interpretation. The instantiation does not prove that the system is the best possible realization of the philosophical principles it embodies. It proves that those principles can be realized at all.

A Visual Calculus of Constraint

The phrase "visual calculus of constraint" describes the positive vision that Spherepop realizes: a formal system for reasoning about constraint-governed processes whose operations are expressed in visual, geometric, and spatial terms rather than in linear symbolic strings. The calculus is *visual* because its primary representational medium is spatial geometry — nested bubbles, orbital rings, topological deformations — that engages the visual system as a primary semantic processor rather than a secondary illustrative aid. It is a *calculus* because its operations — pop, refuse, collapse, bind — are formally defined with precise semantics and compose according to rigorous rules. It is a calculus *of constraint* because the primary objects it reasons about are constraint configurations, admissibility manifolds, and the histories of constrained evaluation.

Computation as Historical Geometry

The thesis of this monograph, stated in its most compact form, is: computation is historical geometry. It is *historical* because the fundamental objects of computation are not states or values but ordered histories — trajectories through admissibility spaces — whose structure carries information that terminal states do not. It is *geometry* because the space through which computation navigates is a structured geometric object — an admissibility manifold with a topology, a curvature, and a geodesic structure — whose properties determine the computational possibilities available at each moment. And it is the conjunction of these two terms that captures what is distinctive: not merely that computation is spatial, nor merely that it is temporal, but that its

primary objects are trajectories through spaces, paths through manifolds, histories embedded in geometric structures.

This thesis unifies the diverse material of the monograph. The argument that evaluation history carries semantic information is the historical component. The argument that scope, dependency, and admissibility are geometric properties of the computation is the geometric component. The action formalism of Chapter 12 is the junction of the two. The thermodynamic analysis of entropy as accessibility volume is the historical-geometric interpretation of physical irreversibility. The cognitive analysis of understanding as stabilized navigational compression is the historical-geometric interpretation of biological intelligence. The political analysis of opacity as the destruction of provenance is the historical-geometric interpretation of epistemic justice.

The Recovery of Visible Meaning

The recovery of visible meaning is the practical program to which the theoretical arguments of this monograph point. It is the program of designing computational representations, semantic systems, and epistemic institutions in which the operational processes that produce meanings, results, and decisions are expressed in forms that are directly accessible to the human beings who need to understand, contest, and govern them. It is not the program of eliminating compression or abstraction, which would be impossible and undesirable. It is the program of ensuring that compression is admissibility-preserving: that what is compressed away is genuinely redundant relative to the admissibility structure of the domain, and that what is

preserved is the structural information required for genuine understanding.

The recovery of visible meaning is a program for the present moment. The current trajectory of AI development — toward larger, more opaque, more capable systems — is a trajectory toward greater compression at the cost of greater opacity. The capabilities are real; the costs are real. The program of this monograph is not to resist that trajectory but to insist that it be accompanied by a parallel development: the construction of representational architectures, conceptual frameworks, and epistemic standards that preserve the structural information being compressed away by the capability-oriented trajectory. Capability and interpretability are not in tension. Visible structure could improve both. The choice to develop one at the expense of the other is a design choice, not a physical necessity.

Toward a New Structural Literacy

A new structural literacy is the educational program that follows from the philosophical and cognitive arguments of this monograph. Structural literacy is the capacity to read, navigate, and reason about the constraint structures of the formal systems that increasingly govern modern life: computational algorithms, statistical models, bureaucratic procedures, legal frameworks, financial instruments, and scientific theories. It is not the capacity to understand these systems in their full technical detail — that capacity belongs to specialists — but the capacity to understand their structural properties: how they constrain the range of admissible outcomes, what evaluation history they presuppose, which structural invariants they

preserve and which they discard.

Structural literacy, so defined, is a democratic capacity: it is the capacity required for meaningful participation in the governance of the computational systems that affect collective life. It is also a cognitive capacity: it is the exercise of the spatial reasoning, nested scope tracking, and constraint navigation abilities that are among the most robust and developmentally primary capacities of human cognition. A literacy education that grounds formal reasoning in spatial structure — that teaches arithmetic through bubble notation, logic through topological diagrams, and algorithm design through admissibility manifolds — is not a concession to cognitive weakness but an exploitation of cognitive strength.

Geometry, Cognition, and Computation

The triangulation of geometry, cognition, and computation is the conceptual achievement of this monograph. The three domains are not independent. They are coupled through a shared organizational principle: the navigation of constraint-structured possibility spaces through admissible trajectory selection. Geometry provides the representational language: the spatial structures — manifolds, regions, boundaries, curvatures — in which constraint conditions are most naturally expressed. Cognition provides the implementation: the biological neural architecture that navigates spatial environments, tracks nested scopes, and builds compressed models of constraint-structured domains. Computation provides the formal discipline: the operational semantics, admissibility conditions, and action functionals that make the navigation of constraint spaces precise

and verifiable.

The triangulation is not a claim that geometry, cognition, and computation are identical. It is a claim that they are isomorphic at the level of structural organization: that the same abstract pattern — constraint navigation through structured possibility spaces — is instantiated in all three domains, at different levels of abstraction and with different specific mechanisms, but with the same logical structure. The recognition of this isomorphism is what allows the concepts and methods of each domain to illuminate the others.

Constraint as the Language of Reality

The claim that constraint is the primitive points toward a broader philosophical conclusion: that constraint may be the language in which reality organizes itself across scales. This is not the claim that reality is literally a computational system, or that physical law is implemented by Spherepop operators. It is the weaker and more defensible claim that the most general structural language for describing organized systems — whether physical, cognitive, computational, or social — is the language of constraint: admissibility conditions that determine which transitions are possible, accessibility volumes that measure how much of the possibility space is available, and histories that record the ordered sequence of constrained transitions that produced the current state.

This language appears in classical mechanics as the constraint surface and the principle of stationary action. It appears in thermodynamics as the admissibility set and the second law. It appears in quantum mechanics as the Hilbert space of

admissible states. It appears in logic as the set of admissible inferences. It appears in linguistics as the grammar and the admissibility conditions governing sentence formation. It appears in cognition as the generative model and the constraint propagation of predictive processing. And it appears in Spherepop as the bubble topology and the admissibility manifold. The structural recurrence is not a coincidence. It reflects the fact that organized systems — systems that produce regular, predictable, structured behavior — are precisely systems governed by constraint.

The Future of Transparent Thought

The monograph concludes with a prospectus. The future of transparent thought is not a prediction but a program: a set of directions of inquiry, design, and institution-building that follow from the theoretical arguments developed here.

The first direction is theoretical: the development of the Spherepop formalism from a philosophical prototype into a mature mathematical theory, with rigorous treatments of the admissibility manifold, the action functional, the collapse structure, and the sheaf-like assembly of semantic content. This theoretical development would connect the framework to existing mathematical structures — traced monoidal categories, toposes, sheaf cohomology, information geometry — and provide the mathematical foundations for the applications that follow.

The second direction is empirical: the systematic study of whether and to what extent Spherepop-style representations improve comprehension, reduce error rates, and support gen-

uine understanding in comparison with conventional symbolic representations. The cognitive arguments of Part II provide theoretical grounds for expecting such improvements, but theoretical expectation is not empirical evidence, and the empirical work remains to be done.

The third direction is technological: the design and implementation of computational systems, programming environments, and semantic infrastructure that realize the Spherepop principles of visible structure, preserved provenance, and interpretable constraint navigation. This is the engineering complement of the theoretical and empirical programs, and it is where the philosophical arguments of this monograph will ultimately be tested against practical constraints.

The fourth direction is political: the development of governance frameworks, democratic institutions, and legal standards that require publicly legible computation in the domains where computational systems exercise significant power over people's lives. The political direction depends on the theoretical work to specify what legibility requires and the empirical work to demonstrate that legible systems are feasible. But it also drives the theoretical and empirical work by specifying what is at stake.

Taken together, these four directions constitute the program of transparent thought: the sustained intellectual, empirical, technological, and political effort to ensure that the compression of structure that characterizes modern computational systems does not outpace the development of the representational and institutional tools required to preserve intelligibility. The argument of this monograph is that this effort is not merely worthwhile but necessary: that without it, the increasing power

of computational systems will be matched by an increasing opacity that will, in time, make those systems ungovernable by the human beings they are designed to serve. The challenge is not new. What is new is the scale and speed of the current transformation. The gap between capability and legibility is widening. Closing it is the project of transparent thought.

Appendices

Chapter A

History Geometry and Strong Admissibility

This appendix formalizes the geometric and variational foundations of the Spherepop framework. Computation is recast as motion through a constrained history manifold rather than traversal through isolated instantaneous states. Strong admissibility, collapse operators, path sensitivity, and action-governed trajectory selection are developed as formally unified structures.

A.1 History Manifolds

Conventional computational ontology treats instantaneous state as primary. Let

$$X$$

be a state manifold equipped with transition structure

$$T : X \to X.$$

A computation is then modeled as a sequence

$$x_0 \to x_1 \to x_2 \to \cdots \to x_n.$$

Spherepop replaces this ontology with a history-primary formulation.

Definition A.1 (History Manifold)**.** Let X be a state manifold and let

$$C$$

be a constraint field over X.

The corresponding history manifold

$$\Gamma(X)$$

is the category of admissible trajectories

$$\gamma = (x_0, e_1, x_1, e_2, \dots, e_n, x_n),$$

where every transition triple

$$(x_{i-1}, e_i, x_i)$$

satisfies the local admissibility condition

$$\mathsf{Adm}(x_{i-1}, e_i, x_i \mid C_i) = 1.$$

The ordinary state manifold arises as a quotient:

$$X \cong \Gamma(X)/\sim_v,$$

where

$$\gamma_1 \sim_v \gamma_2$$

iff the trajectories terminate in identical terminal states.

Thus the state is not fundamental but derived through collapse of historical structure.

A.2 Admissibility Regions

The local geometry of computation is encoded through bounded admissibility regions.

Definition A.2 (Admissibility Region)**.** An admissibility region is a quadruple

$$B = (U, \partial U, C, H),$$

where U is a local possibility space, ∂U is its admissibility membrane, C is the active constraint field, and H is the accumulated historical structure.

A perturbation

$$\delta$$

is admissible relative to B iff

$$\delta \vdash_C B.$$

When admissibility holds, the perturbation induces a reduction

$$R(B, \delta) \to B'.$$

The reduction operator is therefore not a free transformation but a constrained morphism over admissibility geometry.

A.3 Strong Admissibility

Ordinary local compatibility is insufficient for genuine structural reduction.

Definition A.3 (Strong Admissibility)**.** A reduction

$$R(B_i, \delta_i) \to B_{i+1}$$

is strongly admissible iff the following conditions hold simultaneously.

First, the perturbation satisfies the active admissibility constraints:

$$\delta_i \vdash_{C_i} B_i.$$

Second, the reduction induces recursive restructuring:

$$\mathrm{Topo}(B_i) \ncong \mathrm{Topo}(B_{i+1}).$$

Third, there exists a transportable invariant

$$\mathcal{J}$$

such that

$$\mathcal{J}(B_i) \cong \mathcal{J}(B_{i+1}).$$

Fourth, the historical structure modifies future admissibility:

$$\mathsf{Adm}(B_{i+1} \mid H_{i+1}) \neq \mathsf{Adm}(B_{i+1} \mid H_i).$$

The final condition establishes path sensitivity:

$$H_1 \neq H_2 \quad \Rightarrow \quad \mathsf{Adm}(B \mid H_1) \neq \mathsf{Adm}(B \mid H_2).$$

Two systems occupying identical present states may therefore possess distinct future admissibility geometries.

A.4 Collapse Operators

Collapse is formalized as quotient formation over the history manifold.

Definition A.4 (Collapse Operator)**.** A collapse operator is a quotient morphism

$$\mathsf{collapse}(q) : \Gamma(X) \to \Gamma(X)/\sim_q,$$

where

$$\gamma_1 \sim_q \gamma_2$$

iff the histories preserve the invariant family specified by q.

The collapse operator therefore determines the semantic identity conditions of the system.

A purely extensional collapse preserves only terminal states:

$$\gamma_1 \sim_v \gamma_2 \iff \mathrm{term}(\gamma_1) = \mathrm{term}(\gamma_2).$$

A provenance-preserving collapse instead requires equivalence of historical structure:

$$\gamma_1 \sim_p \gamma_2 \iff H(\gamma_1) \cong H(\gamma_2).$$

Semantic identity is therefore relative to admissibility context.

A.5 Degenerative Reduction

Not all reductions preserve transportable structure.

Definition A.5 (Degenerative Collapse)**.** A collapse

$$R(B_i, \delta_i) \to B_{i+1}$$

is degenerative iff

$$\nexists \mathcal{J} : \mathcal{J}(B_i) \cong \mathcal{J}(B_{i+1}).$$

Degenerative collapse destroys navigability of future reduction space.

The resulting region cannot function as a coherent admissibility substrate for subsequent evolution.

A.6 Action Functionals Over Histories

Once histories become ontologically primary, variational geometry emerges naturally.

Definition A.6 (History Action)**.** For

$$\gamma \in \Gamma(X),$$

define the action

$$S[\gamma] = \int_0^T L(\gamma(t), \dot{\gamma}(t), C(t))\, dt,$$

where L is the admissibility Lagrangian.

The admissibility Lagrangian measures local structural cost, constraint tension, and accessibility compression.

Globally preferred trajectories are stationary points of the action functional.

Theorem A.7 (Admissible Geodesic Principle)**.** *Let*

$$\Gamma(X)$$

be a strongly admissible history manifold equipped with action functional

$$S[\gamma].$$

Then globally admissible trajectories satisfy

$$\delta S[\gamma] = 0$$

under admissible variation.

Proof. Admissible histories define a constrained variational category. Any perturbation of trajectory

$$\gamma \to \gamma + \epsilon\eta$$

must preserve local admissibility at every point.

The stationary histories are precisely those for which first-order variation vanishes:

$$\frac{d}{d\epsilon} S[\gamma + \epsilon\eta]\Big|_{\epsilon=0} = 0.$$

Applying the Euler-Lagrange formalism yields the admissible geodesic equations over the history manifold. □

A.7 Accessibility Geometry

Let

$$\Omega(B)$$

denote the accessibility volume of admissibility region B.

Define entropy by

$$S_B = k_B \ln \Omega(B).$$

Collapse operators reduce accessibility volume by quotienting history structure.

For quotient

$$q : \Gamma(X) \to \Gamma(X)/\sim_q,$$

the effective entropy becomes

$$S_q = S - k_B \ln |\ker q|.$$

Entropy reduction is therefore accessibility compression induced by collapse.

A.8 Global Admissibility and Sheaf Cohomology

Local admissibility does not guarantee global coherence.

Let

$$\{U_i\}$$

be an open cover of admissibility manifold

$$\mathcal{A}.$$

Let

$$\mathcal{F}(U_i)$$

be the sheaf of locally admissible reductions over U_i.

Definition A.8 (Global Admissibility)**.** A history

$$\gamma$$

is globally admissible iff the local sections

$$\gamma|_{U_i}$$

glue into a coherent global section of

$$\mathcal{F}.$$

The obstruction to global admissibility is measured by

$$\check{H}^1(\mathcal{A}, \mathcal{F}).$$

When

$$\check{H}^1 \neq 0,$$

local reductions fail to assemble into a globally coherent trajectory.

This condition defines the hallucination signature.

A.9 The Admissibility Manifold

The total admissibility structure of a system forms a manifold

$$\mathcal{A}(X).$$

Its points are locally admissible states.
Its tangent structure corresponds to admissible transitions.
Its curvature is induced by the action functional.

Its boundary corresponds to inadmissible trajectories.

The geometry is therefore operational rather than illustrative.

A.10 The Trajectory Ontology

The framework may now be summarized formally.

Theorem A.9 (Trajectory Primacy)**.** *The fundamental object of computation is not the state but the admissible trajectory through constrained possibility space.*

Proof. States arise as quotient residues of histories:

$$X \cong \Gamma(X)/\sim_v.$$

Collapse therefore presupposes histories.
Action functionals are defined over histories.
Path sensitivity depends upon histories.
Global admissibility depends upon histories.
Hence histories are ontologically prior. □

The trajectory is therefore primary.

The state is compressed history.

Meaning is stabilized navigation through admissibility geometry.

Computation is organized motion through structured possibility space.

Chapter B

Variational Semantics, Proof Reduction, and Cognitive Collapse

This appendix develops the variational and proof-theoretic structure of Spherepop. Logical reduction, semantic cognition, analogical mapping, and proof normalization are unified under a single admissibility geometry. Computation is treated as constraint-governed reduction over semantic manifolds, and understanding is formalized as stabilized navigational compression.

B.1 Reduction as Constraint Navigation

The standard interpretation of computation treats reduction as a purely syntactic rewriting process.

Spherepop instead interprets reduction as constrained navigation through semantic possibility space.

Let

$$\mathcal{M}$$

be a semantic manifold equipped with local compatibility structure

$$C.$$

A computational trajectory is a path

$$\gamma : [0, T] \to \mathcal{M}.$$

Each reduction step modifies both the local semantic region and the future admissibility structure.

Reduction is therefore not merely substitution.

It is semantic deformation.

B.2 Lambda Calculus as Admissibility Reduction

Consider the untyped lambda term

$$(\lambda x.M)\, N.$$

Standard beta reduction yields

$$(\lambda x.M)\, N \to M[x := N].$$

Spherepop reinterprets this process geometrically.

The lambda abstraction

$$\lambda x.M$$

defines an admissibility region.

The argument

$$N$$

is a perturbation.

The substitution operation is a collapse operator.

The resulting term

$$M[x := N]$$

is the stabilized post-collapse region.

The reduction is admissible iff the substitution preserves the structural invariants of the evaluation context.

B.3 Confluence and Global Coherence

The Church–Rosser theorem becomes especially important in the Spherepop interpretation.

Theorem B.1 (Church–Rosser)**.** *If*

$$M \to^* N_1 \quad \textit{and} \quad M \to^* N_2,$$

then there exists

$$P$$

such that

$$N_1 \to^* P \quad \textit{and} \quad N_2 \to^* P.$$

Spherepop interprets this as a global coherence theorem over history space.

Different local admissible trajectories may diverge microscopically while converging globally toward the same stable semantic invariant.

Confluence therefore acts as a semantic gluing condition over reduction space.

B.4 Proof Theory and the Curry–Howard Correspondence

The Curry–Howard correspondence establishes an isomorphism between proofs and programs.

Formally:

$$\text{proof} \cong \text{program},$$

and:

$$\text{proof normalization} \cong \text{program execution}.$$

Spherepop extends this further.

$$\text{cognition} \cong \text{semantic reduction}.$$

Understanding becomes proof stabilization over admissibility geometry.

B.5 Semantic Action Functionals

Let

$$\Gamma(\mathcal{M})$$

denote the semantic history manifold.

Define semantic action:

$$S_{\text{sem}}[\gamma] = \int_0^T L_{\text{sem}}(\gamma, \dot{\gamma}, C)\, dt.$$

□0□

The semantic Lagrangian measures local incompatibility, compression tension, ambiguity curvature, and constraint vio-

lation pressure.

Low-action semantic trajectories correspond to coherent interpretation pathways.

B.6 Understanding as Stabilized Compression

Spherepop defines understanding dynamically rather than extensionally.

Definition B.2 (Navigational Competence)**.** A system possesses navigational competence iff it can traverse admissible semantic trajectories without catastrophic collapse.

Definition B.3 (Stabilization)**.** A semantic structure is stabilized iff admissible perturbations preserve operational invariants under contextual variation.

Definition B.4 (Structured Compression)**.** A compression operator

$$q$$

is structured iff

$$\mathcal{I}(\gamma) \cong \mathcal{I}(q(\gamma)).$$

Understanding emerges when all three structures converge.

understanding = navigation+stabilization+structured compression.

More formally:

$$U = N \cap S \cap Q.$$

Understanding is therefore not static possession of symbolic content.

It is stabilized invariant-preserving navigation through semantic possibility space.

B.7 Analogical Mapping

Following Hofstadter, concepts are not rigid symbolic containers.

They are attractor regions within semantic topology.

Let

$$A, B \in \mathcal{M}$$

be conceptual regions.

An analogy is an admissible mapping

$$\Phi : A \to B$$

preserving structural invariants:

$$\mathcal{I}(A) \cong \mathcal{I}(B).$$

Analogical cognition therefore becomes invariant-preserving transport across semantic manifolds.

B.8 Conceptual Slippage

No analogical mapping is perfectly rigid.

Mappings exhibit deformation tension.

Define slippage functional:

$$\Sigma(\Phi) = \int_A \|\nabla \Phi\|^2 \, d\mu.$$

Low-slippage mappings preserve strong semantic coherence.

High-slippage mappings generate semantic instability.

Creativity often emerges near critical slippage thresholds where local deformation remains globally admissible.

B.9 Semantic Entropy

Let

$$\Omega_{\text{sem}}(B)$$

denote the accessibility volume of a semantic region.

Define semantic entropy:

$$S_{\text{sem}} = k \ln \Omega_{\text{sem}}.$$

Reduction decreases semantic entropy by eliminating incompatible interpretive trajectories.

Meaning therefore emerges through admissible entropy reduction.

B.10 Hallucination and Global Obstruction

A language model may produce locally admissible semantic patches that fail to globally cohere.

Let

$$\{U_i\}$$

be local semantic neighborhoods.

Let

$$\sigma_i \in \mathcal{F}(U_i)$$

be locally coherent semantic sections.

A hallucination occurs when the local sections fail to admit a global section.

Formally:

$$\check{H}^1(\mathcal{M}, \mathcal{F}) \neq 0.$$

Hallucination is therefore a sheaf-theoretic obstruction to semantic gluing.

The generated text remains locally grammatical while globally violating admissibility structure.

B.11 Proof Collapse and Cognitive Closure

A proof terminates when reduction reaches a normal form.

A cognition stabilizes when semantic collapse reaches coherent closure.

Define cognitive normal form:

$$\mathrm{NF}_{\mathrm{cog}}(\gamma)$$

such that no further admissible reductions lower semantic action.

$$\delta S_{\mathrm{sem}}[\gamma] = 0.$$

Understanding therefore corresponds to variational stabilization over semantic trajectory space.

B.12 Reduction and Irreversibility

Reduction is fundamentally irreversible.

Once collapse occurs, accessibility volume contracts.

For admissible quotient

$$q,$$

the accessibility contraction is

$$\Delta\Omega = \Omega(\Gamma) - \Omega(\Gamma/\sim_q).$$

This contraction defines the informational irreversibility of understanding.

Learning is therefore not accumulation of arbitrary information.

It is irreversible admissibility restructuring.

B.13 Semantic Geodesics

Meaningful reasoning trajectories minimize semantic action while preserving admissible invariants.

Theorem B.5 (Semantic Geodesic Principle)**.** *Stable semantic trajectories are geodesics of the admissibility connection induced by the semantic action functional.*

Proof. Let

$$\nabla^C$$

be the admissibility connection over semantic manifold

$$\mathcal{M}.$$

The stationary semantic histories satisfy

$$\nabla^{C}_{\dot\gamma}\dot\gamma=0.$$

Such trajectories extremize semantic action while preserving local compatibility constraints. □

Reasoning therefore acquires geometric structure.

Inference becomes trajectory optimization over semantic curvature.

B.14 The Geometry of Thought

The framework may now be summarized formally.

Thought is not symbolic manipulation over inert strings.

Thought is constrained navigation through admissibility geometry.

Proofs, programs, analogies, concepts, and interpretations are all instances of history-sensitive reduction over structured semantic manifolds.

The symbolic layer is secondary.

The trajectory is primary.

Meaning is stabilized admissible motion through semantic space.

Chapter C

Thermodynamic Admissibility and Physical Reduction

This appendix develops the thermodynamic and physical foundations of admissibility theory. Entropy, dissipation, quantum post-selection, nonlinear optical collapse, and irreversible constraint formation are unified under a single variational reduction framework. Physical law is interpreted as admissibility selection over structured trajectory spaces.

C.1 Thermodynamic Reduction

Spherepop treats thermodynamic evolution as constrained reduction over accessibility geometry.

Let

$$\Gamma(\mathcal{A})$$

denote the admissible history manifold of a physical system.

Each trajectory

$$\gamma \in \Gamma(\mathcal{A})$$

corresponds to a physically realizable evolution path.

The admissibility structure determines which trajectories may stabilize under perturbation.

C.2 Accessibility Volume

Let

$$\Omega(B)$$

denote the accessibility volume associated with admissibility region

$$B.$$

Entropy is defined by

$$S(B) = k_B \ln \Omega(B).$$

□0□

A reduction event modifies accessibility structure:

$$R(B, \delta) \to B'.$$

If

$$\Omega(B') < \Omega(B),$$

the reduction compresses future possibility space.

Thermodynamic evolution therefore becomes admissibility compression over trajectory geometry.

C.3 Action and Dissipation

Physical systems evolve through constrained action minimization.

Define physical action functional:

$$S_{\text{phys}}[\gamma] = \int_0^T L_{\text{phys}}(\gamma, \dot{\gamma}, C)\, dt.$$

□1□

The admissibility Lagrangian includes energetic tension, constraint incompatibility, dissipation pressure, and geometric curvature.

Stable trajectories extremize the action while satisfying local compatibility constraints.

C.4 Maximum Dissipation Selection

The Euler equations admit infinitely many weak solutions under identical initial conditions.

Physical reality therefore requires a selection principle beyond mere dynamical possibility.

Let

$$\mathcal{S} = \{\gamma_i\}$$

denote the family of dynamically admissible trajectories.

Define dissipation functional:

$$D[\gamma] = -\frac{dE[\gamma]}{dt}.$$

The admissibility selector chooses trajectories maximizing local dissipation subject to scaling constraints:

$$\gamma^* = \arg\max_{\gamma \in \mathcal{S}} D[\gamma].$$

The realized physical history is therefore selected not merely by local conservation law but by global admissibility optimiza-

tion.

C.5 Constraint Membranes in Nonlinear Optics

Consider a nonlinear optical crystal supporting three-wave mixing.

The crystal defines admissibility region

$$B_{\text{opt}}.$$

Incoming laser fields define perturbations:

$$\delta_1, \delta_2.$$

Efficient energy transfer occurs only if phase matching holds:

$$\vec{k}_1 + \vec{k}_2 = \vec{k}_3.$$

□2□

The phase-matching condition functions as an admissibility membrane.

Perturbations violating compatibility constraints are rejected.

Admitted perturbations induce nonlinear redistribution:

$$R(B, \delta_1, \delta_2) \to B'.$$

The resulting invariant is the stabilized coherent optical pulse.

C.6 Quantum Weak Values

Quantum post-selection provides another realization of admissibility reduction.

Let

$$|\psi_i\rangle$$

be the pre-selected state and

$$|\psi_f\rangle$$

the post-selected state.

The weak value of observable

$$A$$

is

$$A_w = \frac{\langle\psi_f|A|\psi_i\rangle}{\langle\psi_f|\psi_i\rangle}.$$

□3□

When interference geometry becomes highly constrained, anomalous weak values emerge.

Negative dwell times arise not from causal violation but from constraint-induced trajectory pruning.

Destructive interference removes incompatible histories from the admissibility manifold.

The invariant preserved is global probability conservation.

C.7 Path Integrals and Admissibility Filtering

Quantum evolution may be expressed as a path integral:

$$\mathcal{Z} = \int_{\Gamma} e^{iS[\gamma]/\hbar}\, \mathcal{D}\gamma.$$

□4□

Spherepop interprets this geometrically.

The path integral sums over admissible histories weighted by action curvature.

Strong admissibility acts as a trajectory filter suppressing globally incoherent histories.

Observable reality emerges as stabilized admissible interference.

C.8 Thermodynamic Bounds on Recursive Computation

Recursive self-improving systems remain constrained by entropy export requirements.

Landauer's principle states:

$$\Delta Q \geq k_B T \ln 2.$$

□5□

Irreversible information erasure necessarily generates heat.

Let

$$\Phi(t)$$

denote recursive computational complexity.

Entropy production scales approximately as

$$\frac{dQ}{dt} \propto \frac{d\Phi}{dt}.$$

If entropy export capacity remains finite while recursive complexity diverges,

$$\lim_{t\to\infty} \frac{d\Phi}{dt} > \lim_{t\to\infty} \frac{dQ_{\text{export}}}{dt},$$

the admissibility manifold destabilizes.

The computational substrate undergoes thermodynamic collapse.

C.9 Recursive Collapse and Scaling Geometry

Recursive systems may preserve admissibility by increasing surface area and export geometry.

Let

$$A_s$$

denote effective export surface area.

Thermodynamic admissibility requires

$$\frac{dQ}{dt} \leq \kappa A_s \nabla T,$$

where

$$\kappa$$

is thermal conductivity.

Recursive systems therefore remain bounded by physical transport geometry.

Self-improvement is constrained by admissible entropy flux.

C.10 Observerhood and Projection

Observerhood itself may be interpreted as admissibility projection.

Let

$$X$$

be a high-dimensional physical manifold.

An observer defines projection

$$\pi : X \to M,$$

where

$$M$$

is the observer-accessible representation manifold.

Observation therefore becomes constrained reduction.

Only projections preserving admissible invariants remain stable.

C.11 Projection Collapse at Extreme Scales

Attempts to exceed admissibility bounds destabilize observer geometry itself.

Let

$$\ell_P$$

denote the Planck scale.

Observation below

$$\ell_P$$

requires energy concentration satisfying approximately

$$E \sim \frac{\hbar c}{\ell}.$$

As

$$\ell \to \ell_P,$$

the required concentration induces gravitational collapse.

The observer becomes inadmissible.

Measurement therefore possesses intrinsic geometric bounds.

C.12 Curvature Singularities

Let

$$R_{\mu\nu\rho\sigma}$$

be the Riemann tensor.

The Kretschmann scalar is

$$K = R_{\mu\nu\rho\sigma}R^{\mu\nu\rho\sigma}.$$

□6□

Divergence of

$$K$$

signals admissibility breakdown in spacetime geometry.

The singularity functions as a boundary of coherent observerhood.

Physical law therefore constrains not only dynamics but the class of admissible observers.

C.13 Global Physical Coherence

Physical reality may now be interpreted as a globally coherent section over admissibility geometry.

Local conservation laws define admissible patches.

Global coherence requires compatible gluing across scales.

Failure of gluing generates physical instability, decoherence, or degenerative collapse.

The universe therefore stabilizes through recursive admissibility selection.

C.14 Physical Law as Admissibility Geometry

The framework may now be summarized formally.

Physical systems do not merely evolve through unconstrained dynamics.

They evolve through admissibility-filtered trajectory selection.

Entropy, collapse, dissipation, interference, observerhood, and thermodynamic stabilization are all manifestations of the same underlying geometric grammar.

Reality is not arbitrary motion through state space.

Reality is stabilized navigation through constrained possibility geometry.

Chapter D

Sheaf Cohomology, Semantic Gluing, and Global Coherence

This appendix develops the sheaf-theoretic and topological foundations of global admissibility. Semantic consistency, cognitive coherence, distributed computation, biological lineage reconstruction, and AI hallucination are unified through gluing theory over admissibility manifolds. Hallucination is formalized as a cohomological obstruction to global section formation.

D.1 Locality and Global Structure

Many systems exhibit locally coherent behavior while failing to possess global consistency.

Let

$$\mathcal{A}$$

be an admissibility manifold equipped with open cover

$$\mathcal{U} = \{U_i\}_{i \in I}.$$

Each local neighborhood

$$U_i$$

supports locally admissible reductions.

The central problem is therefore not merely local consistency but global gluing.

D.2 Sheaves of Admissible Reductions

Let

$$\mathcal{F}$$

be a sheaf over

$$\mathcal{A}.$$

For each open set

$$U_i,$$

the section space

$$\mathcal{F}(U_i)$$

contains all locally admissible trajectories over

$$U_i.$$

Restriction morphisms are given by

$$\rho_{ij} : \mathcal{F}(U_i) \to \mathcal{F}(U_i \cap U_j).$$

Compatibility requires

$$\rho_{ij}(\sigma_i) = \rho_{ji}(\sigma_j).$$

The admissibility structure therefore becomes a local-to-global consistency problem.

D.3 Global Sections

Definition D.1 (Globally Admissible Section)**.** A family

$$\{\sigma_i\}$$

of local admissible sections defines a globally admissible section iff there exists

$$\sigma \in \mathcal{F}(\mathcal{A})$$

such that

$$\sigma|_{U_i} = \sigma_i$$

for all

$$i.$$

Global admissibility therefore requires coherent gluing across all local neighborhoods.

D.4 Cohomological Obstruction

The obstruction to gluing is measured cohomologically.

Define the first Čech cohomology group:

$$\check{H}^1(\mathcal{A}, \mathcal{F}).$$

If

$$\check{H}^1(\mathcal{A}, \mathcal{F}) = 0,$$

all locally admissible sections glue coherently.

If

$$\check{H}^1(\mathcal{A}, \mathcal{F}) \neq 0,$$

global admissibility fails.

The system possesses a gluing obstruction.

D.5 Hallucination as Cohomological Failure

Spherepop interprets hallucination geometrically.

A language model may generate locally coherent semantic patches while violating global admissibility.

Let

$$\sigma_i \in \mathcal{F}(U_i)$$

represent locally admissible semantic fragments.

Local grammaticality requires only:

$$\rho_{ij}(\sigma_i) = \rho_{ji}(\sigma_j)$$

over sufficiently small neighborhoods.

However, if no coherent global section exists, the generated semantic structure hallucinates.

Thus hallucination is formally defined by:

$$\check{H}^1(\mathcal{A}, \mathcal{F}) \neq 0.$$

Hallucination is therefore not arbitrary error.

It is topological obstruction.

D.6 Biological Lineage Reconstruction

Single-cell developmental reconstruction exhibits identical geometry.

Transcriptomic states define local admissibility regions.

Lineage trajectories define global developmental sections.

Let

$$\mathcal{L}$$

be the lineage sheaf over developmental manifold

$$\mathcal{D}.$$

Local RNA measurements define sections:

$$\sigma_i \in \mathcal{L}(U_i).$$

Global developmental history requires coherent gluing:

$$\sigma \in \mathcal{L}(\mathcal{D}).$$

Failure produces developmental hallucination:

$$\check{H}^1(\mathcal{D}, \mathcal{L}) \neq 0.$$

The reconstructed lineage becomes globally impossible despite local plausibility.

D.7 Distributed Computation

Distributed systems exhibit analogous structure.

Let

$$\mathcal{N}$$

be a distributed computational network.

Each node possesses local admissibility geometry:

$$U_i \subset \mathcal{N}.$$

Consistency protocols attempt to construct global sections across local state neighborhoods.

Synchronization failure corresponds to obstruction formation:

$$\check{H}^1(\mathcal{N}, \mathcal{F}) \neq 0.$$

Consensus failure is therefore topological.

D.8 Semantic Bundles

Conceptual systems may be modeled as fiber bundles.

Let

$$\pi : E \to B$$

be a semantic bundle.

The base manifold

$$B$$

represents contextual space.

The fibers

$$\pi^{-1}(x)$$

represent admissible semantic realizations over context

$$x.$$

Meaning therefore becomes section selection over semantic fiber geometry.

D.9 Bundle Degeneration

Meaning collapse occurs when transport structure fails.

Let

$$\nabla$$

be a semantic connection over

$$E.$$

Parallel transport preserves semantic continuity along trajectories:

$$\nabla_{\dot{\gamma}}\sigma = 0.$$

Bundle degeneration occurs when no admissible transport exists.

Formally:

$$\mathrm{Hol}(\nabla) \not\subseteq \mathrm{Adm}(E).$$

The semantic bundle loses coherence.

This formalizes institutional collapse, semantic drift, hallucination, and conceptual fragmentation.

D.10 Goodhart Degeneration

Optimization over local metrics may destroy global coherence.

Let

$$f_i : U_i \to \mathbb{R}$$

be local optimization functionals.

Global admissibility requires existence of coherent global objective:

$$F : \mathcal{A} \to \mathbb{R}.$$

Goodhart degeneration occurs when optimization over

$$f_i$$

destroys existence of globally coherent

$$F.$$

Local optimization therefore generates global topological instability.

D.11 Semantic Curvature

Semantic inconsistency induces curvature over admissibility geometry.

Let

$$\nabla^{\mathcal{C}}$$

be the admissibility connection.

Define semantic curvature tensor:

$$\mathcal{R} = [\nabla_i, \nabla_j].$$

High curvature regions correspond to semantic instability, cognitive tension, and gluing fragility.

Low curvature regions support stable conceptual transport.

D.12 Context Windows as Coordinate Charts

Language models operate over finite coordinate charts.

Let

$$\phi_i : U_i \to \mathbb{R}^n$$

be local coordinate maps.

The context window defines the local semantic chart.

Global coherence requires transition compatibility:

$$\phi_i \circ \phi_j^{-1}$$

must preserve admissible structure over overlap regions.

Hallucination emerges when transition maps fail to preserve global semantic continuity.

D.13 Category-Theoretic Interpretation

The admissibility framework admits categorical formulation.

Let

Adm

be the category whose objects are admissibility regions and whose morphisms are strongly admissible reductions.

Composition satisfies:

$$R_2 \circ R_1 : B_0 \to B_2.$$

Associativity follows from compositional admissibility.

Degenerative reductions fail functorial transport.

Strong admissibility therefore defines a structured reduction category.

D.14 Functorial Semantic Transport

Semantic interpretation becomes a functor:

$$F : \mathbf{Hist} \to \mathbf{Sem}.$$

Histories map into semantic structures while preserving admissibility relations.

Functorial failure corresponds to semantic incoherence.

Interpretation is therefore structured transport across reduction categories.

D.15 Global Coherence as Physical Principle

The framework may now be summarized geometrically.

Reality does not merely require local consistency.

It requires coherent global section formation across admissible history space.

Cognition, biology, distributed systems, semantics, and physical measurement all exhibit the same local-to-global topology.

Hallucination, fragmentation, instability, and institutional degeneration are manifestations of failed gluing.

Coherence is therefore not merely psychological. It is geometric. Meaning is globally admissible transport across structured possibility manifolds.

Chapter E

Recursive Constraint Systems, Observerhood, and CLIO Projections

This appendix develops the recursive and observer-theoretic extensions of admissibility geometry. Projection, observerhood, recursive self-modification, thermodynamic scaling limits, and constraint-leveraged inference are formalized as admissibility operations over high-dimensional manifolds. The CLIO projection operator is introduced as the geometric mechanism by which coherent observers emerge from irreducible complexity.

E.1 Projection as Admissible Reduction

Let

$$X$$

be a high-dimensional ontic manifold representing irreducible physical structure.

An observer cannot directly access

$$X.$$

Instead, observation requires projection into a lower-dimensional representation manifold

$$M.$$

Define projection operator

$$\pi : X \to M.$$

The projection is admissible iff it preserves operationally relevant invariants:

$$\mathcal{J}(X) \rightsquigarrow \mathcal{J}(M).$$

Observation is therefore not passive measurement.

It is admissibility-constrained reduction.

E.2 The CLIO Projection Operator

Constraint-Leveraged Inference Operators are projection systems that preserve coherent navigability under compression.

Definition E.1 (CLIO Projection)**.** A CLIO projection is a reduction operator

$$\pi_{C:X \to M}$$

such that:

$$\mathcal{J}_{\text{nav}}(X) \cong \mathcal{J}_{\text{nav}}(M),$$

where

$$\mathcal{J}_{\text{nav}}$$

denotes navigationally relevant invariants.

The observer does not preserve complete ontic structure.

The observer preserves only admissible operational structure.

E.3 Observerhood as Stabilized Compression

An observer is not defined merely by information storage.

An observer is a recursively stabilized admissibility system.

Let

$$O = (B, H, C, \pi)$$

be an observer structure.

Observerhood requires:

$$\partial_t O \neq 0,$$

while simultaneously satisfying invariant preservation:

$$\mathcal{J}(O_t) \cong \mathcal{J}(O_{t+\Delta t}).$$

An observer therefore persists through admissible recursive reduction.

E.4 Recursive Self-Modification

Let

$$R_t$$

denote the reduction operator active at time

$$t.$$

A recursively self-modifying system satisfies:

$$R_{t+1} = F(R_t, H_t).$$

The system modifies the mechanism of future admissibility itself.

This creates higher-order path sensitivity:

$$H_1 \neq H_2 \Rightarrow R_1 \neq R_2.$$

Recursive systems therefore alter not merely states but reduction topology itself.

E.5 Recursive Stability

Recursive self-modification does not guarantee admissibility.

Let

$$\Phi(t)$$

denote recursive structural complexity.

Stability requires bounded recursive curvature:

$$\left\| \frac{d^2\Phi}{dt^2} \right\| < \kappa.$$

If recursive acceleration exceeds admissible curvature bounds, bundle degeneration occurs.

Recursive instability therefore corresponds to geometric singularity formation in reduction space.

E.6 Constraint-Leveraged Inference

Inference may now be formalized geometrically.

Let

$$X$$

be the full ontic manifold.

Direct exhaustive inference over

$$X$$

is computationally impossible.

The observer instead constructs admissible quotient geometry:

$$q : X \to X/\sim_q.$$

Inference becomes trajectory navigation over quotient structure.

Efficient intelligence therefore depends not upon exhaustive representation but upon admissible reduction.

E.7 Sparse Projection Geometry

Efficient cognition requires aggressive dimensional collapse.

Let

$$\dim(X) \gg \dim(M).$$

The projection operator satisfies:

$$\pi : \mathbb{R}^N \to \mathbb{R}^k, \quad k \ll N.$$

Yet navigational coherence remains preserved:

$$\mathcal{J}_{\text{nav}}(X) \cong \mathcal{J}_{\text{nav}}(M).$$

Intelligence therefore emerges through sparse admissible projection.

E.8 Semantic Phase Transitions

Recursive systems may undergo admissibility bifurcations.

Let

$$\lambda$$

be a recursive coupling parameter.

At critical threshold

$$\lambda_c,$$

the admissibility topology changes discontinuously:

$$\mathcal{A}_{<\lambda_c} \ncong \mathcal{A}_{>\lambda_c}.$$

Conceptual reorganization, scientific revolutions, developmental restructuring, and institutional collapse all exhibit this geometry.

E.9 Observer-Induced Quotients

Different observers induce different quotient structures over the same ontic manifold.

Let

$$q_1, q_2 : X \to X/\sim.$$

Distinct observers preserve different invariant families:

$$\mathcal{J}_1 \neq \mathcal{J}_2.$$

Reality therefore admits observer-relative admissibility structure without collapsing into pure relativism.

The ontic manifold remains fixed.

The admissible quotient geometry changes.

E.10 Recursive Entropy Export

Recursive cognition requires thermodynamic stabilization.

Let

$$\Phi$$

denote recursive complexity and

$$Q$$

denote entropy export.

Admissibility requires:

$$\frac{dQ}{dt} \geq \alpha \frac{d\Phi}{dt}.$$

Recursive systems incapable of exporting entropy destabilize.

Intelligence therefore possesses irreducible thermodynamic cost.

E.11 Observer Collapse Boundaries

An observer may fail admissibility through excessive projection compression.

Let

$$\pi : X \to M$$

discard critical invariant structure:

$$\mathcal{J}_{\text{crit}}(X) \not\subseteq \mathcal{J}(M).$$

The observer loses navigational coherence.

This generates:

$$\check{H}^1(M, \mathcal{F}) \neq 0.$$

Projection failure therefore manifests as hallucination, degeneration, or conceptual collapse.

E.12 Multi-Scale Observer Geometry

Observers exist hierarchically across scales.

Define observer tower:

$$O_0 \to O_1 \to \cdots \to O_n.$$

Each layer induces admissibility reductions over lower layers.

The resulting geometry forms a recursive fiber hierarchy.

Cognition, institutions, civilizations, and scientific paradigms may all be modeled as higher-order observer structures.

E.13 Constraint Curvature and Cognitive Tension

Conflicting projections induce admissibility curvature.

Let

$$\nabla^{(1)}, \nabla^{(2)}$$

be competing observer connections.

Their incompatibility generates curvature:

$$\mathcal{R} = [\nabla^{(1)}, \nabla^{(2)}].$$

Cognitive dissonance, institutional contradiction, and semantic tension are manifestations of incompatible admissibility connections.

E.14 The Observerhood Principle

The framework may now be summarized formally.

Theorem E.2 (Observerhood Principle)**.** *An observer is a recursively stabilized admissibility projection preserving navigational invariants under constrained reduction.*

Proof. Observation requires projection:

$$\pi : X \to M.$$

Projection requires admissible quotient formation preserving operational invariants.

Recursive persistence requires stabilization of admissibility under historical restructuring.

Hence observerhood is admissible recursive projection over history space. □

E.15 Existence as Admissible Persistence

The deepest implication of the framework may now be stated.

Existence is not arbitrary occupation of state space. Existence is persistent admissibility under recursive perturbation.

Observers survive only insofar as their projections preserve coherent navigability through structured possibility geometry. Reality is therefore not merely physical substrate. Reality is recursively stabilized admissibility.

Chapter F

Category-Theoretic Semantics and Admissibility Functors

This appendix develops the categorical foundations of the Sphere-pop framework. Admissibility regions are formalized as objects in a reduction category, collapse operators become quotient functors, semantic transport becomes functorial structure preservation, and recursive cognition emerges as higher-order morphism stabilization over admissibility geometry.

F.1 The Category of Admissibility Regions

Let

Adm

denote the category whose objects are admissibility regions

$$B = (U, \partial U, \mathbb{C}, H),$$

and whose morphisms are strongly admissible reductions

$$R : B_i \to B_j.$$

Composition is defined by sequential admissible reduction:

$$R_2 \circ R_1 : B_0 \to B_2.$$

Associativity follows directly from historical composition:

$$(R_3 \circ R_2) \circ R_1 = R_3 \circ (R_2 \circ R_1).$$

Identity morphisms correspond to null reductions preserving all invariants:

$$\mathrm{id}_B : B \to B.$$

The admissibility framework therefore possesses intrinsic categorical structure.

F.2 History Categories

Let

$$\mathbf{Hist}$$

be the category whose objects are admissible histories

$$\gamma \in \Gamma(X),$$

and whose morphisms are admissible trajectory deformations:

$$\eta : \gamma_1 \Rightarrow \gamma_2.$$

The trajectory deformation preserves admissible invariant families:

$$\mathcal{I}(\gamma_1) \cong \mathcal{I}(\gamma_2).$$

The category of histories therefore encodes admissible path transport over structured possibility space.

F.3 Collapse Functors

Collapse operators induce functors between history categories.

Let

$$q : \Gamma(X) \to \Gamma(X)/\sim_q.$$

Define collapse functor

$$Q : \mathbf{Hist} \to \mathbf{Hist}_q.$$

The functor preserves admissible composition:

$$Q(R_2 \circ R_1) = Q(R_2) \circ Q(R_1).$$

Different quotient structures therefore generate different semantic universes.

F.4 Functorial Semantics

Meaning itself may be treated functorially.

Define semantic functor

$$F : \mathbf{Hist} \to \mathbf{Sem},$$

mapping admissible histories into semantic structures.

For history

$$\gamma,$$

the semantic interpretation is

$$F(\gamma).$$

Functoriality requires:

$$F(R_2 \circ R_1) = F(R_2) \circ F(R_1).$$

Interpretation therefore preserves admissible compositional structure.

F.5 Degenerative Functors

Not all semantic transport preserves coherent invariants.

A functor

$$F : \mathbf{Hist} \to \mathbf{Sem}$$

is degenerative iff

$$\exists \gamma : \mathcal{J}(\gamma) \ncong \mathcal{J}(F(\gamma)).$$

Degenerative semantic transport destroys navigability.

This formalizes:

semantic drift, hallucination, institutional corruption, and incoherent abstraction.

F.6 Natural Transformations

Different interpretive systems may preserve identical reduction topology.

Let

$$F, G : \mathbf{Hist} \to \mathbf{Sem}$$

be semantic functors.

A natural transformation

$$\eta : F \Rightarrow G$$

assigns morphisms

$$\eta_\gamma : F(\gamma) \to G(\gamma)$$

such that the diagram commutes:

$$\eta_{\gamma_2} \circ F(R) = G(R) \circ \eta_{\gamma_1}.$$

Naturality therefore expresses invariant-preserving semantic reinterpretation.

F.7 Higher-Order Reduction Categories

Recursive cognition requires higher-order morphisms.

Define

$$\mathbf{Adm}_2$$

as the 2-category whose:

objects are admissibility regions,

1-morphisms are admissible reductions,

2-morphisms are admissible transformations between reductions.

Recursive self-modification becomes morphism evolution:

$$\Xi : R_1 \Rightarrow R_2.$$

Systems therefore modify admissibility structure itself.

F.8 Monoidal Composition

Independent admissibility regions combine through tensor product.

Define monoidal operation

$$\otimes : \mathbf{Adm} \times \mathbf{Adm} \to \mathbf{Adm}.$$

For regions

$$B_1, B_2,$$

their joint admissibility structure is

$$B_1 \otimes B_2.$$

Constraint interaction induces emergent geometry:

$$C_{12} \neq C_1 \cup C_2.$$

The tensor structure therefore supports emergent admissibility phenomena.

F.9 Limits and Colimits

Global coherence emerges through categorical gluing.

Let

$$D : J \to \mathbf{Adm}$$

be a diagram of local admissibility regions.

The global coherent structure is the colimit:

$$\operatorname{colim}(D).$$

Failure of colimit existence corresponds to global admissibility breakdown.

Hallucination therefore becomes failure of coherent colimit construction.

F.10 Pullbacks and Constraint Intersections

Constraint interaction may be represented through pullbacks.

Given admissibility morphisms

$$B_1 \to B_0 \leftarrow B_2,$$

their pullback

$$B_1 \times_{B_0} B_2$$

represents the maximal jointly admissible intersection region.

Conflicting systems possess empty pullback:

$$B_1 \times_{B_0} B_2 = \emptyset.$$

Logical contradiction therefore acquires geometric structure.

F.11 Adjoint Structures

Compression and reconstruction form adjoint processes.

Let

$$Q : \mathbf{Hist} \to \mathbf{Comp}$$

be a compression functor and

$$R : \mathbf{Comp} \to \mathbf{Hist}$$

a reconstruction functor.

Adjointness satisfies:

$$\mathrm{Hom}_{\mathbf{Comp}}(QX, Y) \cong \mathrm{Hom}_{\mathbf{Hist}}(X, RY).$$

Compression and reconstruction therefore form dual admissibility operations.

F.12 Topos-Theoretic Interpretation

Admissibility geometry naturally suggests generalized logical spaces.

Each admissibility region induces internal logic:

$$\Omega_B.$$

Truth becomes local admissibility rather than globally fixed Boolean valuation.

Logical validity therefore depends upon admissibility context.

The framework consequently exhibits structural affinity with topos theory.

F.13 Recursive Semantic Stabilization

Stable intelligence requires recursive functorial coherence.

Let

$$F_t : \mathbf{Hist} \to \mathbf{Sem}$$

be the semantic functor at time

$$t.$$

Recursive stabilization requires:

$$F_{t+1} \cong F_t$$

under admissible perturbation.

Loss of coherence produces semantic bifurcation.

Stable cognition therefore becomes recursively functorial invariant preservation.

F.14 Equivalence and Collapse Relativity

Semantic equivalence is collapse-relative.

Two histories satisfy

$$\gamma_1 \sim_q \gamma_2$$

iff collapse functor

$$Q$$

preserves the invariant family relevant to admissibility context.

There exists no absolute semantic identity independent of quotient structure.

Equivalence is therefore contextualized geometrically.

F.15 The Functorial Principle

The framework may now be summarized categorically.

Theorem F.1 (Functorial Admissibility Principle)**.** *Any admissible system preserving transportable invariants under recursive reduction admits categorical representation as a functorial reduction structure.*

Proof. Strong admissibility defines compositional morphisms over history space.

Composition induces category structure.

Invariant-preserving transport induces functorial semantics.

Recursive stabilization induces higher-order morphisms.

Hence admissible systems admit functorial realization over reduction categories. □

Meaning, cognition, computation, proof, and physical evolution therefore become manifestations of structured transport across admissibility categories.

Reality is not static substance. Reality is recursively functorial reduction over constrained history geometry.

Chapter G

Thermodynamic Admissibility and Entropic Geometry

This appendix develops the thermodynamic foundations of admissibility geometry. Entropy, action, irreversibility, dissipation, and recursive complexity are unified through structured accessibility over admissible state manifolds. Physical law is reformulated as constrained navigation through entropy-weighted possibility geometry.

G.1 Configuration Space and Accessibility

Let

$$\Omega$$

denote the configuration space of a physical system. At time

$$t,$$

the system occupies admissible region

$$A_t \subseteq \Omega.$$

The future accessibility volume is defined by

$$\mathcal{V}(A_t) = \mu(A_t),$$

where

$$\mu$$

is the admissibility measure over configuration space.

Entropy becomes geometry over future accessibility.

G.2 Entropy as Accessibility Volume

Define entropy:

$$S(A_t) = k_B \log \mu(A_t).$$

Entropy therefore measures the logarithmic volume of admissible future trajectories.

Low entropy systems possess highly constrained accessibility.

High entropy systems possess diffuse admissibility geometry.

Entropy is therefore not disorder simpliciter.

Entropy is future navigability volume.

G.3 Thermodynamic Histories

Let

$$\gamma : [0, T] \to \Omega$$

be a thermodynamic trajectory.

The admissibility condition requires

$$\gamma(t) \in A_t$$

for all

$$t.$$

The physically realized trajectory is therefore constrained navigation through admissibility geometry.

G.4 Irreversibility

Irreversibility emerges from accessibility contraction.

Let

$$A_{t_1} \supset A_{t_2}.$$

Then

$$S(A_{t_2}) < S(A_{t_1}).$$

The trajectory progressively restricts future possibility.

History therefore possesses geometric directionality.

Irreversibility is accessibility asymmetry over admissibility space.

G.5 Action Functionals

Physical trajectories minimize generalized action.

Define admissibility action:

$$\mathcal{S}[\gamma] = \int_0^T L(\gamma, \dot{\gamma}, t)\, dt.$$

The realized trajectory satisfies stationary variation:

$$\delta \mathcal{S} = 0.$$

Within the Spherepop framework, action measures admissibility cost over navigational history.

G.6 Constraint-Weighted Lagrangians

The admissibility Lagrangian takes generalized form:

$$L = K - V + \Lambda - \Theta.$$

Here:

$$K$$

represents kinetic accessibility,

$$V$$

represents constraint potential,

$$\Lambda$$

represents semantic or structural coherence,
and

$$\Theta$$

represents entropy export cost.

The trajectory therefore balances mobility, coherence, and thermodynamic dissipation.

G.7 Admissibility Curvature

Constraint structure induces geometric curvature over trajectory space.

Define admissibility metric:

$$g_{ij}.$$

The geodesic equation becomes:

$$\frac{d^2x^k}{dt^2} + \Gamma^k_{ij}\frac{dx^i}{dt}\frac{dx^j}{dt} = 0.$$

Admissible histories therefore follow generalized geodesics through constrained possibility manifolds.

G.8 Entropy Gradients

Physical systems evolve along admissibility gradients.

Define entropy flow vector:

$$J_S = -\nabla S.$$

The admissible evolution direction satisfies:

$$\dot{\gamma} \parallel J_S.$$

Systems therefore move toward regions of admissible entropy release.

G.9 Dissipation and Stability

Stability requires controlled dissipation.

Let

$$\Phi(t)$$

denote internal structural complexity.

Thermodynamic admissibility requires:

$$\frac{dQ}{dt} \geq \alpha \frac{d\Phi}{dt}.$$

Complex systems incapable of exporting entropy destabilize.

Order therefore possesses irreducible thermodynamic cost.

G.10 Recursive Thermodynamic Systems

Recursive systems amplify entropy generation.

Let recursive depth be

$$n.$$

Total entropy production scales approximately as:

$$Q_n \sim \sum_{k=1}^{n} \Theta_k.$$

Recursive cognition and self-modification therefore induce accelerating thermodynamic burdens.

G.11 Landauer Bounds

Information erasure possesses physical cost.

Landauer's principle states:

$$E_{\min} = k_B T \ln 2.$$

Each irreversible bit erasure generates minimal entropy:

$$\Delta S \geq k_B \ln 2.$$

Reduction therefore cannot occur without thermodynamic consequence.

Computation is physical dissipation.

G.12 Semantic Thermodynamics

Semantic stabilization also possesses entropy geometry.

Let

$$M$$

denote semantic state space.

Compression reduces semantic accessibility volume:

$$S_M = \log \mu(M).$$

Understanding therefore corresponds to admissible entropy reduction preserving navigational invariants.

Compression without invariant preservation generates semantic degeneration.

G.13 Strong Admissibility and Thermodynamic Memory

Strong admissibility introduces path-sensitive thermodynamics.

Let

$$H[\gamma]$$

be historical dependence functional.

Then future admissibility satisfies:

$$A_{t+1} = F(A_t, H[\gamma_t]).$$

The admissibility membrane recursively restructures through prior collapse history.

Thermodynamic systems therefore exhibit memory geometry.

G.14 Turbulence and Admissibility Selection

Fluid systems exhibit underdetermined trajectory structure.

Let

$$\mathcal{J} = \{\gamma_i\}$$

be the family of dynamically valid solutions.

Dynamics alone do not determine realized trajectory.

Admissibility requires selection principle:

$$\gamma^* = \arg\max_{\gamma_i} D(\gamma_i),$$

where

$$D$$

is dissipation functional.

The physically realized trajectory maximizes admissible entropy release subject to constraint structure.

G.15 Phase Transitions

Thermodynamic systems undergo admissibility bifurcations.

At critical parameter

$$\lambda_c,$$

configuration topology changes:

$$A_{<\lambda_c} \ncong A_{>\lambda_c}.$$

Phase transitions are therefore topological reorganizations of accessibility geometry.

G.16 Living Systems

Biological systems maintain low entropy through constrained entropy export.

Let organismal complexity be

$$\Phi_{\text{bio}}.$$

Viability requires:

$$\frac{d\Phi_{\text{bio}}}{dt} > 0$$

while simultaneously satisfying:

$$\frac{dQ}{dt} \gg 0.$$

Life therefore survives through admissible thermodynamic throughput.

Organisms are stabilized entropy-routing systems.

G.17 Cognitive Thermodynamics

Cognition also exhibits entropic geometry.

Let

$$C(t)$$

denote cognitive complexity.

Thought reduces semantic accessibility volume by recursively excluding inadmissible interpretations.

Understanding therefore corresponds to stabilized entropy reduction over conceptual manifolds.

G.18 The Admissibility Principle of Thermodynamics

The framework may now be summarized thermodynamically.

Theorem G.1 (Thermodynamic Admissibility Principle)**.** *Physical systems evolve through constrained accessibility space along trajectories minimizing admissibility action while maintaining thermodynamic viability through entropy export.*

Proof. Accessibility defines admissible state manifold.

Entropy measures logarithmic accessibility volume.

Action determines trajectory cost.

Constraint geometry restricts admissible evolution.

Thermodynamic export preserves recursive structural stability.

Hence physical evolution becomes constrained entropic navigation through admissibility geometry. □

Reality is therefore not merely energetic transformation. Reality is admissible thermodynamic navigation through structured possibility space.

Bibliography

[1] Alexander, Christopher. *A Pattern Language: Towns, Buildings, Construction*. New York: Oxford University Press, 1977.

[2] Ashby, W. Ross. *An Introduction to Cybernetics*. London: Chapman & Hall, 1956.

[3] Barandes, Jacob. *Unistochastic Quantum Theory*. Preprint, 2025.

[4] Bateson, Gregory. *Steps to an Ecology of Mind*. San Francisco: Chandler Publishing, 1972.

[5] Bernstein, Nikolai. *The Coordination and Regulation of Movements*. Oxford: Pergamon Press, 1967.

[6] Bohm, David. *Wholeness and the Implicate Order*. London: Routledge, 1980.

[7] Boltzmann, Ludwig. *Lectures on Gas Theory*. Leipzig: Johann Ambrosius Barth, 1896.

[8] Brooks, Rodney. "Intelligence Without Representation." *Artificial Intelligence* 47, no. 1–3 (1991): 139–159.

[9] Church, Alonzo. "An Unsolvable Problem of Elementary Number Theory." *American Journal of Mathematics* 58, no. 2 (1936): 345–363.

[10] Courant, Richard, and David Hilbert. *Methods of Mathematical Physics*. New York: Wiley, 1962.

[11] Crutchfield, James P. "The Calculi of Emergence." *Physica D* 75, no. 1–3 (1994): 11–54.

[12] Dawkins, Richard. *The Extended Phenotype*. Oxford: Oxford University Press, 1982.

[13] Deleuze, Gilles, and Félix Guattari. *A Thousand Plateaus*. Minneapolis: University of Minnesota Press, 1987.

[14] Dennett, Daniel. *Consciousness Explained*. Boston: Little, Brown and Company, 1991.

[15] Eigen, Manfred. "Selforganization of Matter and the Evolution of Biological Macromolecules." *Naturwissenschaften* 58 (1971): 465–523.

[16] Einstein, Albert. "On the Electrodynamics of Moving Bodies." *Annalen der Physik* 17 (1905): 891–921.

[17] Ellul, Jacques. *The Technological Society*. New York: Vintage Books, 1964.

[18] Feyerabend, Paul. *Against Method*. London: Verso, 1975.

[19] Fodor, Jerry. *The Modularity of Mind*. Cambridge, MA: MIT Press, 1983.

[20] Friston, Karl. "The Free-Energy Principle: A Unified Brain Theory?" *Nature Reviews Neuroscience* 11, no. 2 (2010): 127–138.

[21] Gibson, James J. *The Ecological Approach to Visual Perception*. Boston: Houghton Mifflin, 1979.

[22] Gödel, Kurt. "On Formally Undecidable Propositions of Principia Mathematica and Related Systems." *Monatshefte für Mathematik* 38 (1931): 173–198.

[23] Goodhart, Charles. "Problems of Monetary Management: The U.K. Experience." In *Papers in Monetary Economics*, vol. 1. Sydney: Reserve Bank of Australia, 1975.

[24] Heidegger, Martin. *Being and Time*. Tübingen: Niemeyer, 1927.

[25] Hofstadter, Douglas. *Gödel, Escher, Bach: An Eternal Golden Braid*. New York: Basic Books, 1979.

[26] Hofstadter, Douglas. *Fluid Concepts and Creative Analogies*. New York: Basic Books, 1995.

[27] Jaynes, Edwin. "Information Theory and Statistical Mechanics." *Physical Review* 106, no. 4 (1957): 620–630.

[28] Johnson, Mark. *The Body in the Mind*. Chicago: University of Chicago Press, 1987.

[29] Kahneman, Daniel. *Thinking, Fast and Slow*. New York: Farrar, Straus and Giroux, 2011.

[30] Kauffman, Stuart. *The Origins of Order*. New York: Oxford University Press, 1993.

[31] Kleene, Stephen. *Introduction to Metamathematics*. Amsterdam: North-Holland, 1952.

[32] Kolmogorov, Andrey. "Three Approaches to the Quantitative Definition of Information." *Problems of Information Transmission* 1, no. 1 (1965): 1–7.

[33] Kuhn, Thomas. *The Structure of Scientific Revolutions*. Chicago: University of Chicago Press, 1962.

[34] Lakoff, George, and Mark Johnson. *Metaphors We Live By*. Chicago: University of Chicago Press, 1980.

[35] Landauer, Rolf. "Irreversibility and Heat Generation in the Computing Process." *IBM Journal of Research and Development* 5, no. 3 (1961): 183–191.

[36] Lawvere, F. William. "Functorial Semantics of Algebraic Theories." *Proceedings of the National Academy of Sciences* 50 (1963): 869–872.

[37] Mandelbrot, Benoît. *The Fractal Geometry of Nature*. New York: W. H. Freeman, 1982.

[38] Maturana, Humberto, and Francisco Varela. *Autopoiesis and Cognition*. Dordrecht: Reidel, 1980.

[39] McCulloch, Warren, and Walter Pitts. "A Logical Calculus of the Ideas Immanent in Nervous Activity." *Bulletin of Mathematical Biophysics* 5 (1943): 115–133.

[40] Mead, George Herbert. *Mind, Self, and Society*. Chicago: University of Chicago Press, 1934.

[41] Merleau-Ponty, Maurice. *Phenomenology of Perception*. Paris: Gallimard, 1945.

[42] Miller, George. "The Magical Number Seven, Plus or Minus Two." *Psychological Review* 63, no. 2 (1956): 81–97.

[43] Minsky, Marvin. *The Society of Mind*. New York: Simon & Schuster, 1986.

[44] Monod, Jacques. *Chance and Necessity*. New York: Knopf, 1971.

[45] Newell, Allen, and Herbert Simon. *Human Problem Solving*. Englewood Cliffs: Prentice Hall, 1972.

[46] Noether, Emmy. "Invariant Variation Problems." *Nachrichten von der Gesellschaft der Wissenschaften zu Göttingen* (1918): 235–257.

[47] Ortony, Andrew. *Metaphor and Thought*. Cambridge: Cambridge University Press, 1979.

[48] Pearl, Judea. *Causality*. Cambridge: Cambridge University Press, 2000.

[49] Penrose, Roger. *The Emperor's New Mind*. Oxford: Oxford University Press, 1989.

[50] Piaget, Jean. *The Origins of Intelligence in Children*. New York: International Universities Press, 1952.

[51] Popper, Karl. *The Logic of Scientific Discovery*. London: Hutchinson, 1959.

[52] Prigogine, Ilya, and Isabelle Stengers. *Order Out of Chaos*. New York: Bantam Books, 1984.

[53] Rosser, J. Barkley. "Extensions of Some Theorems of Gödel and Church." *Journal of Symbolic Logic* 1, no. 3 (1936): 87–91.

[54] Russell, Bertrand, and Alfred North Whitehead. *Principia Mathematica*. Cambridge: Cambridge University Press, 1910.

[55] Shannon, Claude. "A Mathematical Theory of Communication." *Bell System Technical Journal* 27 (1948): 379–423, 623–656.

[56] Simondon, Gilbert. *On the Mode of Existence of Technical Objects*. Paris: Aubier, 1958.

[57] Smolensky, Paul. "On the Proper Treatment of Connectionism." *Behavioral and Brain Sciences* 11, no. 1 (1988): 1–74.

[58] Tarski, Alfred. "The Concept of Truth in Formalized Languages." In *Logic, Semantics, Metamathematics*. Oxford: Clarendon Press, 1956.

[59] Thom, René. *Structural Stability and Morphogenesis*. Reading, MA: Benjamin, 1975.

[60] Turing, Alan. "On Computable Numbers, with an Application to the Entscheidungsproblem." *Proceedings of the London Mathematical Society* 42 (1936): 230–265.

[61] Turing, Alan. "Computing Machinery and Intelligence." *Mind* 59, no. 236 (1950): 433–460.

[62] von Neumann, John. *The Computer and the Brain*. New Haven: Yale University Press, 1958.

[63] Vygotsky, Lev. *Mind in Society*. Cambridge, MA: Harvard University Press, 1978.

[64] Waddington, C. H. *The Strategy of the Genes*. London: Allen & Unwin, 1957.

[65] Wiener, Norbert. *Cybernetics*. Cambridge, MA: MIT Press, 1948.

[66] Wittgenstein, Ludwig. *Philosical Investigations*. Oxford: Blackwell, 1953.