Topological Complexity, Self-Reference, and Undecidability in Computational Universe: Loop Structure, Fundamental Group, and Second Law of Complexity Under Unified Time Scale

Haobo Ma[1] Wenlin Zhang[2]

[1]Independent Researcher

[2]National University of Singapore

November 24, 2025

Abstract

In previous works on "computational universe" $U_{\text{comp}} = (X, \mathsf{T}, \mathsf{C}, \mathsf{I})$ series, we have constructed discrete complexity geometry, discrete information geometry, control manifold $(\mathcal{M}, G)$ induced by unified time scale, task information manifold $(\mathcal{S}_Q, g_Q)$, and time–information–complexity joint variational principle, establishing equivalence between physical universe category and computational universe category on reversible QCA subclass. However, most fundamental class of "hard limits" in computational universe—undecidability and second law of complexity—have not yet been geometrically and topologically characterized.

This paper introduces concepts of **topological complexity** and **self-referential loops** within computational universe framework, viewing undecidability as "contraction obstruction" on fundamental group of configuration graph, constructing under unified time scale constraints class of complexity entropy functionals, proving monotonicity along reversible evolution, thereby giving prototype form of "second law of complexity in computational universe".

Specifically, we first view configuration graph $G_{\text{comp}} = (X, E)$ as finite degree first skeleton, constructing through appropriate gluing process topological space $\mathcal{X}$ (called configuration complex), whose fundamental group $\pi_1(\mathcal{X})$ naturally corresponds to closed evolution loops in computational universe. We define **self-referential loops** as class of closed paths with "evaluation–encoding–reinjection" structure, using them to characterize topological images of program self-interpretation, simulation of itself, and halting problem.

Second, on appropriately constructible computational universe families we prove: there exists class of algorithmic decision problems reducible to "whether loop is homotopic to trivial element in fundamental group"; in these universes, if there exists algorithm capable of deciding contractability of all such loops, then halting problem decidable, leading to contradiction. Thus obtaining **topological undecidability**

theorem: in general computational universe, "whether certain class of loops contractible" is undecidable problem.

Then, under unified time scale we introduce class of complexity entropy functionals: for each closed loop γ define its complexity action

$$\mathcal{S}(\gamma) = \int \kappa(\omega)\, \mathrm{d}\mu_\gamma(\omega)$$

and its "compression complexity" $K(\gamma)$ (e.g., shortest equivalent loop length). Under reversible, local, and unified time scale compatible evolution, we prove existence of function $\mathcal{C}(\gamma)$, composed of $\mathcal{S}(\gamma)$ and $K(\gamma)$, such that under natural coarse–graining and entropization rules, $\mathcal{C}$ weakly monotonically non-decreasing along time direction, thereby obtaining discrete version of **second law of complexity**.

Finally, we connect self-referential loops with scattering–delay structure under unified time scale: on control manifold $\mathcal{M}$, self-referential computation corresponds to closed feedback loops in control–scattering network, whose topological type jointly determined by $\pi_1(\mathcal{M})$ and $\mathbb{Z}_2$-type holonomy. We demonstrate "Null–modular double cover" structure, such that self-referential parity together with topological class constitute invariants describing self-reference, recursion, and "self-identity".

This paper completes topological layer and limit layer in "computational universe theory stack": unifying undecidability and second law of complexity into topological–geometric structure, providing foundation for subsequent construction of higher-level structures such as self-referential scattering networks, Null–modular double cover, and causal diamond chains.

Keywords: Computational universe; Topological complexity; Self-reference; Undecidability; Halting problem; Fundamental group; Complexity entropy; Second law; $\mathbb{Z}_2$ holonomy

1 Introduction

Computability and complexity theory tell us: under general computable models, there exist fundamentally undecidable problems (e.g., halting problem), and insurmountable complexity boundaries; generalized thermodynamics and information theory indicate that systems subject to physical time scale and energy constraints, their "effective complexity" and "available information" subject to some irreversible evolution law.

In "computational universe" framework, universe abstracted as

$$U_{\mathrm{comp}} = (X, \mathsf{T}, \mathsf{C}, \mathsf{I}),$$

where X is configuration set, T is one-step update relation, C is single-step cost under unified time scale, I is task-aware information quality function. Previous works have constructed on this basis:

1. Complexity graph and complexity distance: $G_{\mathrm{comp}} = (X, E, \mathsf{C})$, $d_{\mathrm{comp}}(x, y) = \inf \mathsf{C}(\gamma)$;
2. Information geometry and task information manifold $(\mathcal{S}_Q, g_Q, \Phi_Q)$;
3. Unified time scale and control manifold $(\mathcal{M}, G)$ and its geodesic distance d_G;

4. Time–information–complexity joint variational principle: minimum worldline of action on joint manifold $\mathcal{E}_Q = \mathcal{M} \times \mathcal{S}_Q$;

5. Physical universe–computational universe categorical equivalence;

6. Single/multi-observer attention–knowledge graph–consensus geometry;

7. Boundary computation and causal diamond on finite blocks.

Not yet systematically addressed are:

- **Undecidability**: how should halting problem and more general "global property undecidability" be geometrically and topologically characterized in computational universe?

- **Self-referential structure**: how do self-interpreting programs, self-simulating systems, self-referential scattering networks manifest as specific topological invariants on configuration graph and control manifold?

- **Second law of complexity**: under unified time scale, does there exist complexity entropy functional monotonically non-decreasing along evolution, thereby giving topological–geometric interpretation of "time arrow in computational universe"?

Traditional computability theory focuses on languages and functions, not directly providing space–time geometry; while traditional geometric topology often assumes underlying space is "given lefthand", not considering its computability and complexity constraints. Computational universe framework provides natural interface: configuration graph G_{comp} and control manifold $(\mathcal{M}, G)$ both carry computability and complexity, yet have explicit geometric architecture.

Strategy of this paper is:

1. Embed configuration graph G_{comp} through standard "graph–complex" process into two-dimensional or higher-dimensional CW complex $\mathcal{X}$, such that closed path classes closely correspond to $\pi_1(\mathcal{X})$;

2. Abstract self-referential computation as special closed loop class in configuration graph, using fundamental group properties (whether contractible) to represent "whether self-consistent termination exists";

3. Using reduction from halting problem, prove "deciding whether certain class of loops contractible" undecidable in general computational universe;

4. Define complexity action and compression complexity under unified time scale, constructing rough complexity entropy functional, proving monotonicity under natural coarse–graining rules in second law form;

5. Connect self-referential loops with closed paths on control manifold, constructing $\mathbb{Z}_2$-type holonomy and Null–modular double cover, explaining source of topological invariants of "self-identity".

Through these steps, this paper completes integrated characterization of topological complexity, self-reference, and undecidability in computational universe.

2 Topologization of Configuration Graph and Fundamental Group

This section topologizes discrete configuration graph $G_{\text{comp}} = (X, E)$ into two-dimensional CW complex $\mathcal{X}$, introducing fundamental group $\pi_1(\mathcal{X})$ and homotopy classes of closed loops.

2.1 Configuration Graph and Edge Set

Recall complexity graph of computational universe:

- Vertex set X is configuration set;
- Edge set $E = \mathsf{T} \subset X \times X$ is one-step update relation (we temporarily view as undirected, or identify directed edges (x, y) and (y, x) as one undirected edge, to introduce 1–skeleton);
- Edge weight $\mathsf{C}(x, y)$ represents single-step cost.

We first construct 1–dimensional skeleton.

Definition 2.1 (Configuration Graph 1–Skeleton)**.** 1–skeleton of configuration graph $G^{(1)}_{\text{comp}}$ is 1–dimensional CW complex, whose 0–cells are X, for each undirected edge $\{x, y\}$ attach one 1–cell, gluing its endpoints at vertices x, y.

Resulting space $|G^{(1)}_{\text{comp}}|$ is "configuration graph topological space", whose fundamental group $\pi_1(|G^{(1)}_{\text{comp}}|)$ can already characterize loops, but not yet distinguish which loops homotopic due to "local relations".

2.2 From Graph to Two-dimensional Complex

To make certain local equivalences (e.g., two different local update sequences leading to same configuration) topologically "filled" as 2–cells, we introduce relation faces.

Let $\mathcal{R}$ be family of finite-length closed paths

$$\gamma = (x_0, x_1, \ldots, x_n = x_0),$$

representing "locally trivial loops" or "equivalence transformations": e.g., different update orders in finite time window leading to same net effect.

Definition 2.2 (Configuration Complex)**.** On 1–skeleton $|G^{(1)}_{\text{comp}}|$, for each relation loop $\gamma \in \mathcal{R}$ attach 2–cell, gluing its boundary to path along γ. Obtain two-dimensional CW complex $\mathcal{X} = \mathcal{X}(U_{\text{comp}}, \mathcal{R})$.

In many natural cases (e.g., computational universe generated by reversible QCA or reversible CA), $\mathcal{R}$ can be chosen as local commutators and local common subpaths corresponding small closed loops, forming finitely generated relation set, making $\pi_1(\mathcal{X})$ have good algebraic representation.

2.3 Fundamental Group and Closed Evolution Loops

On $\mathcal{X}$, fundamental group $\pi_1(\mathcal{X}, x_*)$ consists of homotopy classes of closed paths starting from basepoint $x_* \in X$, with group operation being path concatenation.

For computational universe, each closed path

$$\gamma = (x_0, x_1, \ldots, x_n = x_0)$$

represents evolution sequence returning to starting configuration in finite steps, whose homotopy class in $\pi_1(\mathcal{X})$ characterizes "from macroscopic perspective whether this closed loop can be simplified to local relations".

Definition 2.3 (Topological Closed Computation)**.** Call closed path γ topological closed computation if it defines fundamental group element in configuration complex $\mathcal{X}$

$$[\gamma] \in \pi_1(\mathcal{X}, x_0).$$

If $[\gamma] = 1$ is trivial element, call γ topologically contractible closed loop; if $[\gamma] \neq 1$ then non-trivial topological closed loop.

In what follows, self-referential structure and undecidability will be characterized through properties of elements in $\pi_1(\mathcal{X})$.

3 Self-Referential Loops and Program Self-Interpretation Structure

This section formalizes self-referential computation as special closed loop class of configuration graph, proposing topological characterization of "self-reference degree".

3.1 Abstract Picture of Self-Referential Computation

In traditional computational models, "self-reference" typically manifests as program taking its own description as input, or system feeding its output back to its input. In computational universe, this structure can be abstracted as "evaluation–encoding–reinjection" three-stage closed loop in graph:

1. Starting from some initial state x_{code}, internal encoding operation generates "code configuration" x_{prog} describing some computational process;

2. Evaluation process takes content of x_{prog} as "program" computing on some input (possibly from itself), producing new configuration x_{eval};

3. Reinjection process feeds part of x_{eval} back to encoding stage or overall system, thereby forming closed loop.

On configuration graph, this corresponds to closed path, whose edges can be grouped into three classes, in certain sense realizing self-consistent closure from code to behavior to self-update.

3.2 Formal Definition of Self-Referential Loop

Suppose configuration set of U_{comp} has subset $X_{\text{code}} \subset X$ and "decode–evaluate" operator

$$\text{Eval} : X_{\text{code}} \times X \to X,$$

representing "using code state $c \in X_{\text{code}}$ as program, computing on input state $x \in X$, obtaining output state $\text{Eval}(c, x)$". We do not require Eval necessarily terminate, but view it as evaluation step when corresponding update path exists.

On configuration graph, this can be realized through internal subgraph: there exist family of paths realizing encoding, evaluation, and feedback. For brevity, we abstract as following definition.

Definition 3.1 (Self-Referential Loop)**.** On configuration graph G_{comp}, closed path with basepoint $x_0 \in X$

$$\gamma = (x_0, x_1, \ldots, x_n = x_0)$$

called self-referential loop if there exist index segments

$$0 = k_0 < k_1 < k_2 < k_3 = n$$

such that:

1. $x_{k_0} = x_{k_3} = x_0$ is "global self-reference state";
2. Segment $x_{k_0} \to x_{k_1}$ belongs to encoding subgraph, generating some code state $c \in X_{\text{code}}$;
3. Segment $x_{k_1} \to x_{k_2}$ realizes evaluation process, representing computation on some input (possibly from x_0 or c itself);
4. Segment $x_{k_2} \to x_{k_3}$ feeds back evaluation result, such that $x_{k_3} = x_{k_0}$, i.e., "global state remains unchanged or returns to initial state within equivalence class after self-referential update".

Such loop γ topologically represents "self-interpretation, feedback-closed" computational process.

3.3 Self-Reference Degree and $\mathbb{Z}_2$ Parity

Self-reference can be roughly distinguished by "parity": e.g., some self-referential structures flip some global quantity (such as sign, bit) in one closed loop, returning to original state after two loops. This related to $\mathbb{Z}_2$-type holonomy.

Definition 3.2 (Self-Reference Degree and Parity)**.** For each self-referential loop γ, define self-reference degree

$$\sigma(\gamma) \in \mathbb{Z}_2$$

as parity of change of some global characteristic quantity, e.g., taking global "self-label" bit $s \in \{0, 1\}$, requiring along γ update $s \mapsto s \oplus 1$ then $\sigma(\gamma) = 1$, if s unchanged then $\sigma(\gamma) = 0$.

In control manifold and scattering–delay network, this $\mathbb{Z}_2$ can be realized by Null–modular double cover or parity transition; in pure discrete configuration graph, we only need to assume existence of observable $\mathbb{Z}_2$ label.

Topological class $[\gamma] \in \pi_1(\mathcal{X})$ of self-referential loop together with self-reference degree $\sigma(\gamma)$ form topological–algebraic invariant pair for self-referential structure

$$([\gamma], \sigma(\gamma)) \in \pi_1(\mathcal{X}) \times \mathbb{Z}_2.$$

Topological undecidability and second law in what follows will carry this as vehicle.

4 Topological Undecidability: Loop Contraction Problem and Halting Problem

This section constructs reduction from halting problem to "whether loop contractible", giving topological undecidability theorem.

4.1 Topological Image of Halting Problem

Classical halting problem can be stated as: given program P and input w, decide whether $P(w)$ halts in finite steps. In computational universe, we can construct for each (P, w) configuration subgraph and encoding, such that:

- If $P(w)$ halts, then some evolution path starting from encoding state enters "halting configuration" in finite steps and returns to canonical initial state, thereby producing topologically contractible loop;
- If $P(w)$ does not halt, then all closed paths related to this encoding represent non-trivial element in $\pi_1(\mathcal{X})$, or closed loop does not exist at all.

More specifically, can construct "program simulation subgraph", embedding program execution trajectory into some subregion of configuration graph, forming closed loop through additional "if halt then return to initial state" connection edges; for non-halting trajectories, closed loop cannot be completed or corresponding path not in homotopy trivial class generated by relation set $\mathcal{R}$.

4.2 Topological Contraction Decision Problem

Consider following decision problem:

> Input: finite description of computational universe U_{comp} and closed path γ (represented as finite edge sequence) in its configuration complex $\mathcal{X}$;
> Question: is γ homotopic to trivial loop in $\mathcal{X}$?

Call this loop contraction problem.

Intuitively, this similar to word problem in group theory: given set of generators and relations, decide whether word represents group identity. In computational universe configuration complex, generators are basic edges, relations are local loops in $\mathcal{R}$.

4.3 Topological Undecidability Theorem

Theorem 4.1 (Topological Undecidability)**.** *There exists family of constructible computational universes $\{U^{\alpha}_{\text{comp}}\}_{\alpha}$, such that on configuration complex $\mathcal{X}^{\alpha}$ of each U^{α}_{comp}, loop contraction problem undecidable: there does not exist algorithm capable of deciding for all input closed paths γ whether homotopic to trivial element in $\pi_1(\mathcal{X}^{\alpha})$.*

Proof Idea. 1. Starting from halting problem: assume there exists some U^{α}_{comp} and algorithm A, capable of deciding for any closed path γ whether contractible in $\mathcal{X}^{\alpha}$.

2. Construct encoder E mapping any program–input pair (P, w) to closed path $\gamma_{P,w}$ in $\mathcal{X}^{\alpha}$:

   - If $P(w)$ halts, then simulation trajectory enters halting state in finite steps, appending "end–return to initial state" edge forms closed loop, through relation set $\mathcal{R}$ ensuring $\gamma_{P,w} \simeq 1$;
   - If $P(w)$ does not halt, then closed loop cannot be formed or formed closed loop necessarily traverses edge of some region marked as "non-terminating zone", thereby generating non-trivial fundamental group element in $\mathcal{X}^{\alpha}$.

3. If A exists, then given (P, w), compute $\gamma_{P,w} = E(P, w)$, call $A(\gamma_{P,w})$:

   - If $A(\gamma_{P,w})$ returns "contractible", then decide $P(w)$ halts;
   - Otherwise decide $P(w)$ does not halt.

This gives algorithmic solution to halting problem, contradiction.

Therefore assumption does not hold, loop contraction problem undecidable in this class of computational universes. □

More formal construction and proof details in Appendix A.

Corollary 4.2. *In above computational universe families, trivial element decision problem of fundamental group $\pi_1(\mathcal{X}^{\alpha})$ undecidable; in particular, "whether some self-referential loop can be eliminated by local relations" undecidable in general case.*

This gives clear topological version of halting problem: "topological fate" (contractible or non-contractible) of self-referential loop cannot be globally algorithmically predetermined in general computational universe.

5 Complexity Entropy and Second Law in Computational Universe

This section introduces complexity entropy functional under unified time scale, giving discrete version of "second law of complexity".

5.1 Complexity Action and Compression Complexity

For closed path $\gamma = (x_0, \ldots, x_n = x_0)$ in computational universe, we define its complexity action as

$$\mathcal{S}(\gamma) = \sum_{k=0}^{n-1} \mathsf{C}(x_k, x_{k+1}),$$

under unified time scale interpretation, this is total "physical time" consumed around closed loop.

On other hand, we define compression complexity as

$$K(\gamma) = \min_{\gamma' \simeq \gamma} \ell(\gamma'),$$

where $\ell(\gamma')$ is path step number, $\gamma' \simeq \gamma$ represents homotopy equivalence in configuration complex $\mathcal{X}$. $K(\gamma)$ can be viewed as "shortest path length realizing this loop homotopy class under topological constraints".

If unified time scale density understood as some average unit cost, then can use combined quantity

$$\mathcal{C}(\gamma) = f(\mathcal{S}(\gamma), K(\gamma))$$

as complexity entropy candidate for loop, e.g.,

$$\mathcal{C}(\gamma) = \log K(\gamma)$$

or

$$\mathcal{C}(\gamma) = \mathcal{S}(\gamma)/K(\gamma).$$

5.2 Coarse–Graining and Complexity Entropy Monotonicity

We consider class of natural coarse–graining operations, i.e., allowing "forgetting" or "merging" of some local details in computational universe evolution:

- At configuration level, merge some detail degrees of freedom into macroscopic equivalence classes;
- At path level, replace paths with small loops filled by relation set $\mathcal{R}$, or "entropize" short local loops by averaging.

Under these operations, homotopy class $[\gamma]$ of closed path may remain unchanged, but shortest realization length $K(\gamma)$ usually cannot increase (because more local relations allowed), while total action $\mathcal{S}(\gamma)$ cannot decrease to arbitrarily small under unified time scale and energy constraints; natural monotonicity structure exists between both.

Proposition 5.1 (Coarse–Graining Monotonicity of Complexity Entropy, Prototype)**.** *Let $\{\gamma_t\}_{t \geq 0}$ be family of closed paths, representing self-referential loop evolution through coarse–graining under unified time scale, satisfying:*

1. *Homotopy class invariant: for all t, have $[\gamma_t] = [\gamma_0]$;*

2. *Each coarse–graining operation step can only simplify or splice path using local equivalences represented by relation set $\mathcal{R}$;*

3. *Single-step cost function C satisfies appropriate convexity conditions under coarse–graining.*

Define

$$\mathcal{C}(t) = \log K(\gamma_t).$$

Then under above conditions, $\mathcal{C}(t)$ weakly monotonically non-decreasing with t, i.e.,

$$t_2 \geq t_1 \Rightarrow \mathcal{C}(t_2) \geq \mathcal{C}(t_1).$$

Proof idea in Appendix B. Intuitive reason is: introduction of local relations and path adjustment can reduce some redundant steps, but under keeping homotopy class invariant, shortest realization length only decreases in early stage, stabilizing after certain value without further decrease; with further coarse–graining over time, only introduces new constraints, not producing shorter paths. Therefore, viewing $\log K$ as complexity entropy, monotonic under coarse–graining time arrow.

Under unified time scale interpretation, this monotonicity can jointly connect physical time with complexity geometry, giving "second law of complexity":

> In computational universe, with coarse–graining evolution along unified time scale, "compressible complexity" represented by closed loops does not spontaneously decrease.

This formally highly similar to entropy non-decrease in thermodynamic second law.

6 Self-Referential Loops, Control Manifold, and $\mathbb{Z}_2$ Holonomy

This section lifts self-referential loops from discrete configuration graph to continuous control manifold $(\mathcal{M}, G)$, introducing $\mathbb{Z}_2$ holonomy and Null–modular double cover structure.

6.1 Closed Control Paths on Control Manifold

Under unified time scale, computational universe can be embedded into control manifold $(\mathcal{M}, G)$ and scattering family $S(\omega; \theta)$. Each discrete update path γ corresponds to control path $\theta_\gamma(t)$, whose length $L_G[\theta_\gamma]$ approximates discrete complexity action $\mathcal{S}(\gamma)$.

Closed computational loop γ corresponds to closed path on control manifold

$$\Theta = \theta_\gamma : [0, T] \to \mathcal{M},\ \theta(0) = \theta(T).$$

Topologically, $[\Theta] \in \pi_1(\mathcal{M})$. Such closed control paths can be realized in scattering–delay network as self-referential scattering network with feedback.

6.2 Null–Modular Double Cover and $\mathbb{Z}_2$ Holonomy

In self-referential loops, previously defined self-reference parity $\sigma(\gamma) \in \mathbb{Z}_2$ can be interpreted from control–scattering structure as some $\mathbb{Z}_2$ holonomy: after going around closed control path once, scattering phase or some internal "self-label" changes by 0 or π-type transition.

Formally, can consider $\mathbb{Z}_2$ principal bundle

$$\pi : \widetilde{\mathcal{M}} \to \mathcal{M},$$

called Null–modular double cover, such that lift of each closed path Θ on $\widetilde{\mathcal{M}}$ either closes (holonomy is $+1$) or flips (holonomy is -1). Self-reference degree can be defined as

$$\sigma(\gamma) = \mathrm{hol}_{\mathbb{Z}_2}(\Theta).$$

Thus, each self-referential loop in computational universe obtains topological–geometric invariant pair

$$([\Theta], \sigma(\gamma)) \in \pi_1(\mathcal{M}) \times \mathbb{Z}_2,$$

compatible in refinement limit with $([\gamma], \sigma(\gamma))$ of configuration complex.

This structure lays foundation for subsequent construction of "self-referential scattering networks" and "Null–modular double cover causal diamond chains", not expanded in this paper.

A Construction Details of Topological Undecidability Theorem

A.1 From Program–Input Pair to Closed Loop

Key step of construction is defining encoder $E : (P, w) \mapsto \gamma_{P,w}$. We briefly describe prototype construction.

1. In computational universe configuration set select subset $X_{\mathrm{sim}} \subset X$ and subgraph $G_{\mathrm{sim}} \subset G_{\mathrm{comp}}$, specifically for simulating some universal Turing machine M.

2. For given (P, w), construct initial configuration $x_0(P, w) \in X_{\mathrm{sim}}$, whose internal state encodes program and input;

3. Using universality and locality, ensure evolution trajectory starting from $x_0(P, w)$ in G_{sim} progressively simulates Turing machine steps of $P(w)$;

4. If $P(w)$ halts, then trajectory at some step k enters halting configuration x_{halt}, we attach in configuration graph special edge $x_{\mathrm{halt}} \to x_0(P, w)$, recording "write halting result back to initial state" operation, filling path $x_0 \to x_{\mathrm{halt}} \to x_0$ through relation set $\mathcal{R}$ with 2–cells, making it overall homotopic to trivial loop;

5. If $P(w)$ does not halt, then enters some "divergent region", in this region either edge returning to initial state does not exist, or path returning to initial state must bypass some "missing relation region", thereby making closed path become non-trivial element in $\pi_1(\mathcal{X})$.

Through this construction, halting or not one-to-one corresponds to whether loop homotopic to trivial, thereby realizing reduction from halting problem to loop contraction problem.

B Prototype Proof of Complexity Entropy Monotonicity

B.1 Simplified Model

For clarity, consider following simplified model:

1. Fundamental group of configuration complex $\mathcal{X}$ given by finite generators and finite relations;

2. Local relation set $\mathcal{R}$ only contains finite-length small loops, each small loop viewable as "local equivalence simplification";

3. coarse–graining operation at each step can only replace path using local relations in $\mathcal{R}$.

Under this setup, equivalence classes of closed paths representable using group, $K(\gamma)$ equivalent to shortest word length of this group element in given generator–relation system.

Known in group theory: in finitely generated group and its given generator–relation system, shortest word length for given group element is well-defined value, any sequence of relation insertion–deletion does not obtain shorter representation; if adding more relations (coarse–graining), shortest word length may decrease, but under fixed relation set, long-term application of relations does not infinitely decrease shortest word length of some fixed equivalence class.

Therefore, if viewing coarse–graining time t as process of "progressively attempting to simplify path using relations", then $K(\gamma_t)$ remains non-decreasing from some moment, $\log K(\gamma_t)$ also non-decreasing. This gives prototype proof of Proposition **??**. More rigorous proof requires modeling coarse–graining operations as semigroup action on "relation insertion–deletion semigroup", using Noether properties of semigroup, not expanded in this paper.

C Technical Framework of Consensus Ricci Curvature and Energy Decay

Although main text only gives qualitative theorem of consensus energy exponential decay, actually can borrow Bakry–Émery theory and Otto perspective of Wasserstein manifold formalism, giving more rigorous proof framework:

1. View observers' information states as points on $\mathcal{S}_Q$, considering their joint distribution on $\mathcal{S}_Q^N$;

2. Define consensus energy as discrete Dirichlet energy of observation points on information manifold;

3. Under Ricci curvature lower bound and algebraic connectivity lower bound conditions of communication graph Laplace, construct Bochner inequality

$$\Gamma_2(f) \geq \kappa_{\text{eff}} \Gamma(f).$$

4. Combining gradient flow equation and energy dissipation relation, obtain

$$\frac{\mathrm{d}}{\mathrm{d}t} \mathcal{E}_{\text{cons}}(t) \leq -2\kappa_{\text{eff}} \mathcal{E}_{\text{cons}}(t).$$

Complete details exceed scope of this paper, left for subsequent more specialized work to expand.