

Unified Recursion Theory (URT): Core Constraint Framework

Author: Morgan Baggs

Role: Composite foundational architecture

Status: Canonical constraint document

Scope: Architectural synthesis only

Excludes: Derivations, numerics, validation, datasets, dynamics, geometry

Updated: Jan 30 2026

Scope and Role of This Document

This document defines the **core constraint architecture** of Unified Recursion Theory (URT).

It synthesizes the foundational constraint content of:

- **Foundation I:** *Minimal Admissibility of Irreversible Updates*
- **Foundation II:** *Canonical Operator Semantics and Constraint Closure in Unified Recursion Theory*
- **Foundation III:** *Oscillation Recursion Mirror: Admissibility Evaluation Under Competing Irreversible Updates in Unified Recursion Theory*

This document **does not reproduce** derivations, proofs, datasets, numerics, or validation results. All such material appears exclusively in the above foundations and downstream papers.

URT is presented here **solely as a constraint theory** governing the admissibility of irreversible physical updates.

1. Canonical Vocabulary and Primitive Objects

URT operates on informational descriptions of physical systems. The following objects are **primitive**.

1.1 Informational State

An informational state I_n represents the configuration and accessible microstates of a physical system at recursion index n .

No assumptions are made about spacetime dynamics or temporal evolution.

1.2 Informational Entropy Change ΔH

ΔH quantifies the magnitude of **informational compression** associated with an update:

- $\Delta H = 0$: entropy-preserving update
- $\Delta H > 0$: irreversible update

1.3 Energetic Cost ΔE

ΔE is the energetic expenditure required to realize an irreversible informational update.

URT constrains **whether** such an expenditure is admissible; it does not model how it is supplied.

1.4 Environmental Bandwidth T

T denotes the local environmental capacity supporting informational processing.

Its physical realization is domain-dependent; its role is universal.

1.5 Informational Stiffness σ

σ quantifies **structural resistance** to irreversible informational reconfiguration.

- Structural descriptor
- Independent of efficiency diagnostics
- Domain-specific in realization, universal in role

1.6 Efficiency Diagnostic λ

λ is a **process-class dependent diagnostic** quantifying the efficiency of irreversible updates.

- λ is **computed** from the admissibility relation between energy expenditure and informational entropy change
- λ is **not prescribed or optimized**
- λ is implied by the relation between energy and informational change

Formal process-class structure and closure: *Canonical Operator Semantics and Constraint Closure in Unified Recursion Theory*.

2. Admissible Update Modes

URT distinguishes **constraint classes**, not dynamics.

2.1 Conserving Updates Ψ_{cons}

- $\Delta H = 0$
- Entropy-preserving
- Reversible limit

No efficiency diagnostic is evaluated.

2.2 Compressive Updates Ψ_{comp}

- $\Delta H > 0$
- Entropy-producing
- Irreversible

Compressive updates are subject to admissibility constraints. URT does not specify their dynamical realization.

3. Minimal Admissibility Constraint (Foundation I)

All irreversible updates must satisfy the universal admissibility relation:

$$\Delta E = \lambda k_B T \Delta H$$

Where λ is **process-class dependent**:

- Conserving processes: $\lambda \geq 1$
- Compressive processes: $\lambda < 1$

This relation constrains **possibility**, not efficiency optimization.

Formal derivation, proof, and falsification criteria appear **only** in *Minimal Admissibility of Irreversible Updates*.

4. Canonical Operator Semantics and Closure (Foundation II)

URT operates on a **closed, non-redundant operator set**:

$$\Delta E, \Delta H, T, \sigma, \lambda, \Psi_{\text{cons}}, \Psi_{\text{comp}}$$

This set is:

- **Complete**: sufficient to describe all irreversible updates

- **Non-redundant:** no operator can be eliminated
- **Closed:** no additional primitives are required

Operators:

- do **not** generate dynamics
- do **not** encode preferences
- do **not** rank outcomes

Their role is **structural classification of admissible updates**.

Formal closure proof: ***Canonical Operator Semantics and Constraint Closure in Unified Recursion Theory***.

5. Informational Stiffness in Admissibility (Foundation III)

5.1 Role of Stiffness

Stiffness enters admissibility through the dimensionless ratio:

$$\frac{\sigma}{k_B T}$$

- High $\sigma/(k_B T)$: compression structurally suppressed
- Low $\sigma/(k_B T)$: compression structurally permitted

Interpretive representation (where applicable):

- σ may be represented as free-energy curvature $\partial^2 F / \partial H^2$ in thermodynamic contexts (Foundation II)

Stiffness is **not** an efficiency parameter and does not encode dynamics.

6. Oscillation Recursion Mirror (ORM)

ORM is the **admissibility evaluator** in URT.

6.1 ORM Function

ORM performs **binary filtering**:

- inadmissible updates are removed

- admissible updates remain available

ORM never selects, ranks, optimizes, or controls outcomes.

6.2 ORM Constraint Structure

Admissibility requires simultaneous satisfaction of:

Energetic consistency

$$\Delta E = \lambda k_B T \Delta H$$

1. **Environmental sufficiency**

T adequate for the specified ΔH

2. **Structural feasibility**

$\sigma/(k_B T)$ within **process-class admissible bounds** (defined in **Foundation III**)

Failure of any constraint renders an update inadmissible.

7. Competing Irreversible Updates

When multiple irreversible updates are dynamically possible:

- ORM filters inadmissible candidates
- Remaining admissible updates are resolved externally

Resolution may involve dynamics, stochasticity, or boundary conditions. URT does not participate in this resolution.

8. Architecture-Level Falsification

URT is falsified if:

- Irreversible updates persist while violating admissibility
- ORM must be reinterpreted as a controller or optimizer
- Informational stiffness cannot be coherently defined across domains

Composite architecture failure:

If maintaining consistency requires:

- modifying operator definitions
- introducing new primitive operators
- reinterpreting ORM as dynamics or control

then the URT constraint architecture fails.

Empirical falsification appears only in downstream papers.

9. Program Dependency Summary

Core Architecture

This core framework is defined exclusively by the following foundational papers:

- **Foundation I:** *A Minimal Admissibility Constraint for Irreversible Physical Updates*
- **Foundation II:** *Canonical Operator Semantics and Constraint Closure in Unified Recursion Theory*
- **Foundation III:** *Oscillation Recursion Mirror: Admissibility Evaluation Under Competing Irreversible Updates in Unified Recursion Theory*

These papers define all admissibility constraints, operator semantics, and evaluation rules used by URT.

Downstream Realization

All realizations of URT occur in separate downstream papers, which apply the core constraints without modifying them. These include:

- **Diagnostic structure of efficiency and stiffness**
(λ and σ treated as empirical or inferred quantities)
- **Geometric and structural interpretations**
(informational geometry, free-energy landscapes)
- **Field-level and strong-constraint extensions**
(interpretive treatments of curvature and stiffness fields)
- **Domain-specific applications**
(quantum systems, biology, cosmology, spacetime structure, particle physics)
- **Consistency realization protocols**
(tiered analyses verifying admissible regions without predictive claims)

All downstream papers are required to remain consistent with the constraints defined in this core framework and may not redefine operators or admissibility condition

10. Conclusion

URT is a **constraint architecture**, not a dynamical theory.

It unifies physical domains by specifying the conditions under which **irreversible informational change is possible at all**.

Dynamics explain *how* systems evolve.

Unified Recursion Theory defines *where evolution is admissible*.