Milestone 010 — What state should a reasoner preserve?

Tyler J.H.G.

Abstract

When is it safe for a recurrent reasoner to treat two states as equivalent?

Research period: September 2–3, 2026
Historical anchors: #193, #198, #223

This branch started with a question about intermediate language: would translating text into something more structurally explicit, such as Lojban, help a model reason about it?

To evaluate that idea, I first needed to specify what the translation was supposed to preserve. Tokenization, translation, and semantic normalization perform different operations; changing the serialization gives us no automatic guarantee about the information a later reasoning step will need.

Suppose two situations have different descriptions but produce the same outcomes for every task we care about. We might represent them as the same state and save the cost of preserving their differences. Now let the reasoner update both states. If their future behavior diverges, that initial merger has discarded something the recurrence needed.

The question became: **when can a reasoner safely treat two states as equivalent?**

1. Visual intuition

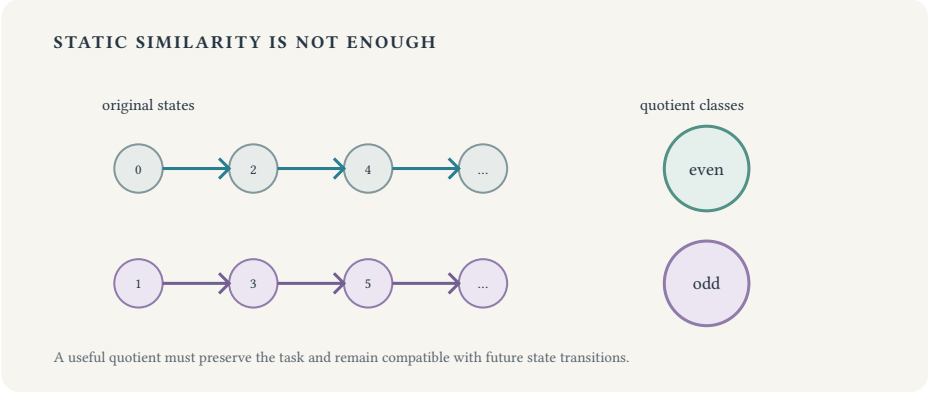

Figure 1: A toy quotient groups states by parity; adding two preserves the equivalence classes.

Published Sep 29, 2026

2. From static to dynamic sufficiency

A compressed recurrent state needs to satisfy two requirements. It must preserve the distinctions required by the task, and its updates must continue to respect those distinctions.

Consider two states that look equivalent now. If updating both keeps them equivalent under every relevant future observation, we can continue treating them as the same. If an update exposes a difference that matters later, the original equivalence was too coarse.

> **Sticky note — quotient:** group objects by an equivalence relation, then treat each group as one object for the purpose being studied. Defining the relation specifies which distinctions we are prepared to discard.

This gave me a target for the representation: a **dynamic or predictive quotient**, retaining enough state to preserve the distinctions required for relevant future behavior. Keeping every detail can be expensive and can leave irrelevant variables available as shortcuts. Compression becomes useful when we can specify why the discarded differences will remain irrelevant.

3. Research context

Several established theories address versions of this question.

A **sufficient statistic** preserves the information needed for a specified inference. The information bottleneck studies a related tradeoff between information retained about an input and information useful for a target.

Sequential systems add the requirement about future behavior. **Predictive State Representations** describe state through predictions of observable futures. **Causal-state** constructions in computational mechanics group histories that imply the same distribution over futures. Myhill–Nerode equivalence and bisimulation likewise specify when states can be merged while preserving the future behavior under consideration.

I arrived at the recurrence-compatible equivalence question before doing this literature sweep. The sweep gave it a more precise vocabulary and several mature constructions to work with. The next task was to use those existing notions of predictive sufficiency to constrain a practical representation.

4. Try a small example

A six-state toy system groups states by parity. Adding two preserves each state's class.

```
# Toy quotient: states are equivalent if they have the same parity.
states=range(6)
```

```
R=lambda x:(x+2)%6  # respects parity
for x in states:
    print(x, 'class', x%2, '->', R(x), 'class', R(x)%2)
```

Saved output:

```
0 class 0 -> 2 class 0
1 class 1 -> 3 class 1
2 class 0 -> 4 class 0
3 class 1 -> 5 class 1
4 class 0 -> 0 class 0
5 class 1 -> 1 class 1
```

5. Why Lojban stopped being the main question

Lojban, AMR, DRS, UCCA, or another structured representation could expose relationships that ordinary text leaves implicit. Their usefulness depends on which distinctions the system needs and which the compiler preserves.

If a compiler maps two inputs to the same state, a downstream reasoner receiving only that state has no way to distinguish them. That is acceptable for a task whose outcomes treat them as equivalent. A later task that depends on the discarded difference changes the requirement.

I therefore treated each candidate language as a possible serialization or approximation of the desired state. Its evaluation needed a behavioral test: which inputs does it merge, and can any relevant future operation distinguish them?

6. Interpretation

The quotient is defined relative to a family of future observations, actions, or tasks. Choosing that family is part of the modeling problem; unknown future requirements limit how confidently we can discard information.

The formulation gives a proposed semantic representation a concrete test. Identify the distinctions it merges, then check whether the relevant reasoning dynamics preserve that equivalence.

7. Sources and chronology

- Milestone notebook 010 — preserved chronological source
- Research chronology
- Publication dispositions and Atlas-to-article map

The matching Atlas theme is **Dynamic state equivalence**. The earlier Visual Intuition Atlas notebook preserves the source visualization.

The next chronological article is **Milestone 011 — Geometry should follow invariance, not aesthetics**.