Compositional Behavioral Semantics for State Abstraction in Reinforcement Learning

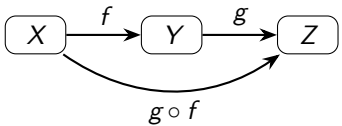
Yivan Zhang, Ray Luo, Manuel Baltieri

July 24, 2026

Category: Objects and Structure-Preserving Maps

A category has:

- objects;
- morphisms/arrows between objects;
- identity arrows;
- composition of arrows (when the endpoints match).



Generalizes settings like:

sets/functions, vector spaces/linear maps.

Why needed

Category theory is the language for understanding relations and structures between objects.

The Category **Set**: State Spaces and Functions

In this paper, the main category is usually **Set**:

$$\text{objects} = \text{sets}, \qquad \text{morphisms} = \text{functions}.$$

- State spaces are sets:

  S (represent concrete state space in RL), Z (abstract state space in RL), X (generic space).

  In category theory, objects are secondary; their relationships through morphisms are primary.
- Encoders are morphisms:

  $$\phi : S \to Z.$$
- Transition functions are morphisms:

  $$t_S : S \to FS.$$

  They map each state to its one-step behavior.

Functor: Mapping Categories (Objects and Morphisms)

A functor F maps:

$$X \mapsto FX, \qquad f : X \to Y \mapsto Ff : FX \to FY.$$

It preserves identities and composition:

$$F(\mathrm{id}_X) = \mathrm{id}_{FX}, \qquad F(g \circ f) = Fg \circ Ff.$$

System Type: What Counts as One-Step Behavior?

In the paper, a functor specifies the **type of one-step behavior** (input-output interface).

Stochastic Moore machine functor

$$F_{\text{Moore}}X := (\mathcal{P}X)^{\mathcal{A}} \times \mathcal{O}.$$
$$F_{\text{Moore}}f := (\mathcal{P}f)^{\mathcal{A}} \times \text{id}_{\mathcal{O}}.$$

Read this as:

- for each action $a \in \mathcal{A}$, a distribution over next states;
- plus an observation/output in $\mathcal{O}$ (deterministic for simplicity; with a reward),.
- formulates how the environment in RL (POMDP) unfold.

Closed-loop (PO) Markov process (composing policy + env)

$$F_{\text{Markov}}X := \mathcal{P}X \times \mathcal{O}.$$
$$F_{\text{Markov}}f := \mathcal{P}f \times \text{id}_{\mathcal{O}}.$$

Stochastic Moore Functor: Elementwise View

In other words, an element of $F_{\mathrm{Moore}}X$ is

$$(p : \mathcal{A} \to \mathcal{P}X,\ o \in \mathcal{O}).$$

Given a map $f : X \to Y$,

$$F_{\mathrm{Moore}}f : F_{\mathrm{Moore}}X \to F_{\mathrm{Moore}}Y$$

maps

$$(p, o) \longmapsto (f_* p, o),$$

where

$$f_* p : \mathcal{A} \xrightarrow{p} \mathcal{P}X \xrightarrow{\mathcal{P}f} \mathcal{P}Y.$$

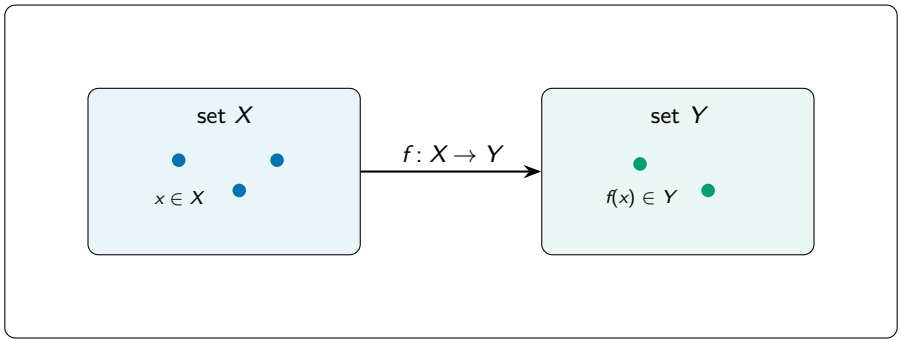
Thus, for each action $a \in \mathcal{A}$,

$$(f_* p)(a) = (\mathcal{P}f)(p(a)) \in \mathcal{P}Y.$$

- The transition distribution is pushed forward through f.
- The observation o is unchanged.

From Elements to Sets to Categories

Set: category of sets and functions



elements live inside sets; sets are objects in **Set**; functions are morphisms

Coalgebra: A State Transition System that Unfolds Dynamics

Simple formulation

A state transition system can be seen as an F-coalgebra which is a pair (X, t_X), where:

$$t_X : X \to FX,$$

and F is an endofunctor. In **Set**, this means a set X equipped with a map t_X.

More explicitly:

- $F : \mathcal{C} \to \mathcal{C}$ is an endofunctor.
- X is an object of $\mathcal{C}$.
- FX is the object of one-step behaviors over X.
- $t_X : X \to FX$ is a morphism in $\mathcal{C}$.
- In this paper, $\mathcal{C} = \mathbf{Set}$, so X is a state set and t_X is an ordinary function that maps one state space to another. FX is still a set whose element $\in (\mathcal{P}X)^{\mathcal{A}} \times \mathcal{O}$.

Algebra: A Structure that Aggregates

Simple formulation

An F-algebra is a pair (A, a), where:

$$a : FA \to A.$$

In **Set**, this means a set A equipped with a map a.

More explicitly:

- $F : \mathcal{C} \to \mathcal{C}$ is an endofunctor.
- A is an object of $\mathcal{C}$.
- FA is the object of structured A-data.
- $a : FA \to A$ is a morphism in $\mathcal{C}$.
- Algebraic operators aggregate or evaluate structure, e.g. $\wedge$, $+$, expectation, and almost-sure aggregation.

coalgebra: $X \to FX$ unfolds behavior

algebra: $FA \to A$ aggregates structure

Coalgebra Homomorphism: Dynamics Preservation

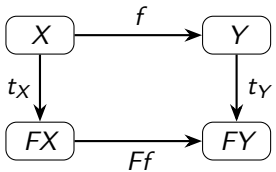
Given systems

$$t_X : X \to FX, \qquad t_Y : Y \to FY,$$

a map $f : X \to Y$ is a homomorphism if

$$t_Y \circ f = Ff \circ t_X.$$

This says: encode first or transition first; the result agrees. ("diagram commutes": the composite functions are the same)



Why needed

This is the formal condition behind homomorphic abstraction.

Homomorphic Abstraction

Given a concrete system

$$t_S : S \to FS$$

and an encoder

$$\phi : S \to Z,$$

ϕ is a homomorphic abstraction if there exists

$$t_Z : Z \to FZ$$

such that

$$t_Z \circ \phi = F\phi \circ t_S.$$

Why needed

This is the central abstraction criterion in the paper: the representation must preserve one-step dynamics (again, reward as part of observation), i.e., how state space expands through transition.

Moore Homomorphism in RL Terms

For Moore systems, homomorphism expands to:

$$t_Y(\phi(x), a) = (\mathcal{P}\phi)(t_X(x, a)), \qquad o_Y(\phi(x)) = o_X(x).$$

- Abstract transitions are induced by concrete transitions after applying ϕ.
- Observation/reward is preserved (same outcomes no matter which underlying dynamics is used).
- This is studied as bisimulation abstraction / self-predictive abstraction (RP+ZP) in RL.
- Sidenote: In RL bisimulation literature, there is a mess in describing this, but in their style I can describe as "all bisimulation abstraction not limited to the largest one that map to the smallest state space."

Natural Transformation: Changing System Type

For functors F, G, a natural transformation

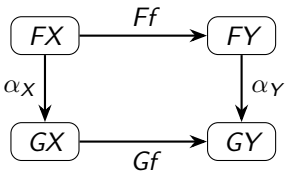
$$\alpha : F \Rightarrow G$$

is a family of maps

$$\alpha_X : FX \to GX$$

such that transforming first or mapping first gives the same result:

$$Gf \circ \alpha_X = \alpha_Y \circ Ff.$$



Why needed

It formalizes converting one kind of system into another, e.g., closing an environment with a policy. Objects are connected by morphisms; categories are connected by functors; functors are connected by natural transformations.

Policy induces a Natural Transformation

A policy

$$\pi : \mathcal{O} \to \mathcal{A}$$

turns a Moore system into a closed-loop Markov system:

$$\alpha^{\pi} : F_{\text{Moore}} \Rightarrow F_{\text{Markov}}.$$

For each X:

$$\alpha^{\pi}_{X} : (\mathcal{P}X)^{\mathcal{A}} \times \mathcal{O} \to \mathcal{P}X \times \mathcal{O}$$

is

$$\alpha^{\pi}_{X}(p, o) = (p(\pi(o)), o).$$

Why needed

This lets us describe policy-dependent transition compositionally.

Bundle: Local Behavioral Semantic

Definition

An n-ary bundle over value set V is a map

$$h_X : X^n \to V.$$

Examples:

$$p : X \to \{\top, \bot\} \quad v : X \to \mathbb{R} \quad r : X \times X \to \{\top, \bot\} \quad d : X \times X \to [0, \infty].$$

Why needed

Bundles are the local, state-dependent behavioral semantics. Semantics is what things mean.
Can equip additional structure on V, e.g., an order that allows bundle comparison.

Bundle Lifting: One-Step Semantics

A bundle lifting maps a bundle on states to a bundle on one-step behaviors.

Informal formulation

Given a bundle

$$h_X : X^n \to V,$$

a lifting constructs a bundle

$$h_{FX} : (FX)^n \to V.$$

In the notation of the paper, we write this operation as a map

$$h_{FX} = \lambda_X(h_X).$$

- h_X describes a behavioral semantic on states.
- h_{FX} describes how that semantic is evaluated on one-step behaviors (by varying operators).

Example 3.3: Safety Property

Let

$$V = \{\top, \bot\}, \qquad h_O : \mathcal{O} \to \{\top, \bot\}$$

be a safety predicate on observations.

Lifted safety predicate

For a unary bundle $h_X : X \to \{\top, \bot\}$,

$$h_{FX}(p, o) := (\forall_{x \sim p} h_X(x)) \wedge h_O(o).$$

- $h_O(o)$: the current observation is safe.
- $\forall_{x \sim p} h_X(x)$: any next state that can be sampled is safe.
- You can think of p as the next state distribution $p(s'|a)$, which needs to take in an action/dist. over action. An element of FX is (p, o).
- This gives a one-step local form of safety preservation.

Example 3.4: Reward Aggregation

Let

$$V = \mathbb{R}, \qquad h_O : \mathcal{O} \to \mathbb{R}$$

be a reward or scalar cumulant on observations.

Lifted reward aggregation

For a unary bundle $h_X : X \to \mathbb{R}$,

$$h_{FX}(p, o) := \gamma \, \mathbb{E}_{x \sim p}[h_X(x)] + h_O(o),$$

where $\gamma \in [0, 1]$ is a discount factor.

- $h_O(o)$: immediate reward/cumulant.
- $\gamma \mathbb{E}_{x \sim p}[h_X(x)]$: discounted future quantity.
- Under a fixed policy, this becomes the Bellman-style update.

Example 3.5: Successor Feature

Suppose the scalar cumulant factors through features:

$$h_O : \mathcal{O} \xrightarrow{\varphi} \Phi \xrightarrow{w} \mathbb{R},$$

where $\Phi = \mathbb{R}^d$.

Vector-valued aggregation

For a vector-valued unary bundle $h_X^\Phi : X \to \Phi$,

$$h_{FX}^\Phi(p, o) := \gamma \, \mathbb{E}_{x \sim p}[h_X^\Phi(x)] + \varphi(o).$$

- h_X^Φ tracks future feature accumulation.
- $\varphi(o)$ is the immediate feature observation.
- Applying w recovers the scalar reward aggregation:

$$w(h_{FX}^\Phi(p, o)) = h_{FX}(p, o).$$

Pullback: Move Semantics along f on System Y to System X

Given

$$f : X \to Y, \qquad h_Y : Y^n \to V,$$

define pullback operation f^* that produce a new bundle on X^n, namely $f^* h_Y$:

$$(f^* h_Y)(x_{1:n}) = h_Y(f(x_1), \ldots, f(x_n)).$$

Meaning

The bundle originally lives on Y, and along f we transport the bundle back to X.

Pullback Example: Isometric Embedding

This shows an example of pullback operator that moves semantics from an abstract to a concrete system. Let $\phi : S \to Z$ be an encoder, and let

$$d_Z : Z \times Z \to [0, \infty]$$

be a metric on the representation space Z.

Pullback of the representation metric

$$\phi^* d_Z : S \times S \to [0, \infty], \qquad (\phi^* d_Z)(s, s') = d_Z(\phi(s), \phi(s')).$$

Isometry condition

$$d_S = \phi^* d_Z, \qquad \text{i.e.} \quad d_S(s, s') = d_Z(\phi(s), \phi(s')).$$

- Pullback moves back the representational metric back to the raw state space.
- Isometry condition: compare the pull-backed distance and the behavioral distance. The abstraction (ϕ) preserves the behavioral distance on concrete state space.

Closure Operator: Pull One-Step Semantics Back to States

Start with a state-level bundle:

$$h_X : X^n \to V.$$

Step 1: Lifting

The lifting turns h_X into a bundle on one-step behaviors:

$$\lambda_X(h_X) = h_{FX} : (FX)^n \to V.$$

Step 2: Pullback along dynamics

The system map

$$t_X : X \to FX$$

sends each state to its one-step behavior, so we pull h_{FX} back along t_X:

$$T_X(h_X) := t_X^*(h_{FX}) = t_X^*(\lambda_X(h_X)).$$

Equivalently,

$$T_X(h_X)(x_{1:n}) = h_{FX}(t_X(x_1), \ldots, t_X(x_n)), \;\; T_X = t_X^* \circ \lambda_X.$$

Intuition: What Does the Closure Operator Do?

The closure operator asks:

If h_X describes semantics on states, what does one step of the system say h_X should become?

- h_X: current candidate behavioral semantics on states.
- $\lambda_X(h_X)$: how to evaluate that semantics on one-step behaviors.
- t_X^*: substitute each state x by its actual one-step behavior $t_X(x)$.
- $T_X(h_X)$: the one-step update of h_X induced by the system.

Connection to Bellman updates

For reward aggregation, T_X becomes the Bellman operator: current reward plus discounted expected future value.
Lifting: takes h_X and defines how to aggregate it over one transition.
Pullback: evaluate the lifted semantics at the actual behavior $t_X(x)$ of each state x.

Behavioral Structure: Stability Under Update

A fixed point satisfies:

$$h_X = T_X(h_X).$$

A post-fixed point satisfies:

$$h_X \preceq T_X(h_X).$$

- Value functions are fixed points of Bellman-like operators.
- Bisimulation relations can be post-fixed points of binary closure operators.

Why needed

This is how the paper defines when a bundle is a genuine behavioral structure for a system.

Pushforward: Move Concrete Structures to Abstract States

Given

$$f : X \to Y, \qquad h_X : X^n \to V,$$

define pushforward operation f_*:

$$f_* h_X(y_{1:n}) = \bigvee_{x_i \in f^{-1}(y_i)} h_X(x_{1:n}).$$

Meaning

Pushforward aggregates concrete behavior over the states represented by each abstract state. Meaning of join operation $\bigvee$ depends on V and its order (e.g., max function, logical OR, in general, least upper bound) assuming we have such a join operation.

Why needed

It lets the paper construct abstract behavioral structures from concrete ones.

How the Concepts Fit Together

Role	Concept	Paper object
state spaces, encoders	**Set** (obj, arrow)	$S, Z; \phi : S \to Z$
system type	functor F	$F_{\mathrm{Moore}}, F_{\mathrm{Markov}}$
one-step dynamics	coalgebra	$t_X : X \to FX$
abstraction criterion	homomorphism	$t_Z \circ \phi = F\phi \circ t_S$
behavior semantics	bundle	$h_X : X^n \to V$
transition-level bundle	lifting	$h_X \mapsto h_{FX}$
one-step bundle update	closure operator	$T_X = t_X^* \circ \lambda_X$
transfer along encoder	pullback/pushforward	ϕ^*, ϕ_*

Section 3.5: What Do We Get?

We now have two systems and an abstraction map:

$$\phi : (S, t_S) \to (Z, t_Z).$$

Goal

Transfer behavioral structures between the concrete state space S and the abstract state space Z.

- Theorems 3.12 and 3.13 say behavioral structures transfer across ϕ forward and backward, when ϕ is an homomorphic abstraction.

Theorem	Operation	Intuition
3.12	Pullback ϕ^*	Abstract behavioral structure remains valid when viewed on concrete states.
3.13	Pushforward ϕ_*	Concrete behavioral structure can be summarized into a valid abstract one.

Theorem 3.12: Safe Verification

Suppose

$$\phi : (S, t_S) \to (Z, t_Z)$$

is a **surjective homomorphism**.

Immediate result

If the pulled-back bundle $\phi^* h_Z : S^n \to V$ is a behavioral structure on the concrete system, then $h_Z : Z^n \to V$ is a behavioral structure on the abstract system:

$$\phi^* h_Z \preceq T_S(\phi^* h_Z) \quad \Longrightarrow \quad h_Z \preceq T_Z(h_Z).$$

- Start with an abstract claim h_Z.
- Pull it back to concrete states: $\phi^* h_Z$.
- If the concrete system validates it, then the abstract system validates it.
- Surjectivity means Z has no abstract states unsupported by S.

Theorem 3.13: Safe Construction

Suppose

$$\phi : (S, t_S) \to (Z, t_Z)$$

is a **homomorphism**.

Immediate result

Any concrete behavioral structure $h_S : S^n \to V$ can be pushed forward to a valid abstract one:

$$h_S \preceq T_S(h_S) \quad \Longrightarrow \quad \phi_* h_S \preceq T_Z(\phi_* h_S).$$

- Start with a concrete structure h_S.
- Aggregate it over each abstract fiber (inverse image):

$$(\phi_* h_S)(z_{1:n}) = \bigvee_{s_i \in \phi^{-1}(z_i)} h_S(s_{1:n}).$$

- The result is automatically stable under the abstract dynamics.
- No surjectivity is needed for this direction.

Why This Framework Is Meaningful

The contribution is not isolated transfer theorems.
It is a reusable recipe for defining and proving abstraction guarantees.

Specification recipe

$$\text{bundle arity} + \text{codomain} + \text{lifting operators} \implies \text{behavioral structure}$$

- **Bundle arity**: state safety property, value, relation, metric, higher-order semantics.
- **Codomain**: Boolean truth values, real values, distances, ordered quantities.
- **Lifting operators**: how one-step transitions aggregate the state-level semantics to transition-level semantics.

Once this structure is specified, the transfer theorems say how it behaves and transfer under homomorphic abstraction.

Future Directions

The rebuttal frames the paper as a foundation for designing new abstraction tools.

- **Agent state abstraction**: abstract internal agent states, not only environment states.
- **Approximate guarantees**: replace strict homomorphisms by lax homomorphisms.
- **Beyond strict homomorphic abstraction**: preserve chosen behavioral structures without preserving the full system.
- **Behavioral analysis beyond reward**: compatible for the analysis of env/agent's behavior from perspectives other than mere reward: safety, temporal logic, and neurosymbolic objectives, or new abstraction objectives like new behavioral metrics.