

The Geometry of Scope

Context, Visibility, and Translation Across Formal Systems

Flyxion

Independent Researcher

October 2026

Abstract

Formal languages, operating systems, computational environments, and interactive virtual worlds employ different mechanisms for organizing context, visibility, and permissible operations. Despite their differences, many admit representations in terms of nested or overlapping domains, locally available information, transition relations, and structure-preserving mappings. This essay develops a mathematical framework for comparing such systems without presupposing their equivalence. The primitive construction is a contextual representation system, consisting of a small category of contexts, a set of underlying states, a presheaf of context-indexed observations coherent with the restriction arrows, an assignment of admissible actions, and a partial transition function. Knowledge, presentation, and authority are shown to be independent assignments, so that hierarchical containment alone forces no monotonicity of visibility or permission, and navigation between contexts is separated from disclosure within a context by an explicit commutation criterion. Filesystems, shell environments, lexical scope, access control, and game worlds are treated as instances, and the orientation of information flow is identified as the structural feature that distinguishes sections over regions from environments built by accumulation. For every system, a canonical presheaf of observational distinctions is constructed and characterized by a universal property, the coarsest quotient of the state space compatible with a chosen family of contexts is characterized in the same way, and the terminal system observed through a fixed observation presheaf is identified with its global sections. Failures of local-to-global reconstruction are classified and exhibited by finite examples. Translations between systems are then analyzed by what they preserve, reflect, collapse, or invent: observational distinctions are preserved by every translation and reflected exactly under an injectivity condition, authority is preserved only in a lax direction that discards prohibitions, and surjectivity on states yields a common abstraction without yielding equivalence. A representation theorem for tree-shaped systems shows that a filesystem and a lexical scope discipline arise from one decorated rooted tree by dual product constructions. The essay closes by separating a common abstraction from categorical equivalence, from translation of proof theories, and from computational universality, and argues that equivalence of heterogeneous formal systems is always equivalence relative to the structure that a chosen abstraction retains.

Contents

I	The Mathematics of Context	3
1	Introduction: Representation and Structural Correspondence	3
2	Contexts as Mathematical Objects	5
3	Scope, Visibility, and Authority	7
4	Navigation and Disclosure as Distinct Operators	9
II	Hierarchical and Computational Representations	11
5	Filesystems and Shell Environments	11
6	Lexical Scope and Abstract Syntax Trees	13
7	Access Control and Contextual Authority	15
8	Interactive Worlds as Transition Systems	16
III	Presheaves and Local Information	18
9	Presheaf Semantics for Contextual Observation	18
10	Local Consistency and Global Reconstruction	19
11	Hidden Information and Observational Equivalence	21
IV	Functorial Translation	24
12	Categories of Representational Systems	24
13	Common Abstractions and Forgetful Functors	25
14	Surjectivity, Fullness, Faithfulness, and Equivalence	29
15	Translation Between Mathematical Languages	31
16	Universality and Its Limits	33

V	Results and Interpretation	34
17	A General Contextual Translation Construction	35
18	Preservation and Loss Under Translation	38
19	The Limits of Hierarchical Representation	40
20	Conclusion: Equivalence Relative to Abstraction	42
A	Definitions and Notation	44
B	Finite Examples and Counterexamples for Visibility, Disclosure, and Access Control	44
C	Worked Translations Among Filesystem, Lexical-Scope, and Transition-System Models	45
D	Formal Inventory: Propositions, Counterexamples, and Open Questions	47
E	Related Frameworks and Bibliographic Notes	49

Part I

The Mathematics of Context

1 Introduction: Representation and Structural Correspondence

Programs, filesystems, shells, access-control policies, and the interactive worlds of video games share a recurring design problem. At any moment a participant occupies some context, and the context determines what can be seen, what can be named, what can be done, and where one can go next. A block of source code determines which identifiers resolve and to what. A directory determines which entries are listed and which paths are interpreted relative to it. A process determines an environment, a working directory, and a set of credentials. A level, a menu, or a room in an interactive world determines which objects are present, which controls are live, and which exits are open. The recurrence is striking enough to invite sweeping formulations, in which all of these are said to be trees, or hierarchies, or instances of a single structure, and in which the further step is sometimes taken that all mathematical languages are therefore equivalent. The purpose of this essay is to treat the recurrence as a mathematical question while declining the sweeping formulations, and to determine by

definitions, theorems, and counterexamples how much common structure there is and what each comparison discards.

Four claims need to be kept apart. The first is that two systems exhibit analogous organizational patterns, which is a report of resemblance and carries no mathematical content until the resemblance is made precise. The second is that the two systems admit mappings into a common abstract structure. The third is that those mappings preserve specified mathematical properties, such as reachability among contexts, distinctions among states, or restrictions on action. The fourth is that the systems are equivalent under a specified notion of equivalence. Only the first two are available at the outset, and the second is weaker than it appears, because every system maps to a one-point system in which nothing is distinguished and nothing is forbidden, so that the existence of a common abstraction is trivial unless the abstraction is required to retain something. The third and fourth claims require explicit conditions and proofs, and the results below are organized around the conditions under which each can be established and the counterexamples that show where each fails.

The central research question is the following: under what conditions can heterogeneous formal systems be represented by a shared structure of contexts, observations, and admissible transformations? The answer developed here has a definite form. Relative to a specified family of observable structures, distinct systems possess a canonical common abstraction, namely a coarsest quotient of states that retains exactly the distinctions the family can detect, characterized by a universal property rather than selected by convenience. The differences between the systems are then characterized by what that abstraction necessarily forgets: the multiplicity of microstates behind each observable profile, the dynamics, the authority structure, and the orientation in which information flows between nested contexts. The thesis is therefore neither that all such systems are equivalent nor that nothing can be said in common, but that equivalence is always equivalence relative to a retained structure, and that the retained structure can be named, constructed, and proved canonical.

Three levels of claim are distinguished throughout, and each numbered result carries its level. The first level consists of established results from category theory, sheaf theory, logic, and programming-language theory that serve as foundations; these are cited rather than reproved, except where a short proof clarifies how a general fact applies to contextual systems. The second level consists of results proved here, among them the commutation criterion for navigation and disclosure, the canonical presheaf with its universal properties, the terminal observed system, the relative equivalence theorem, the duality between regional and accumulative information flow, the representation theorem for tree systems, the lax treatment of authority, and the analysis of what is lost when a graph is replaced by a hierarchy. The third level consists of conjectures and open questions, which are labeled as such. Appendix D inventories every result with its status. No claim is made concerning human cognition; whether people habitually employ these abstractions is an empirical question outside the scope of the mathematics.

The contribution is positioned as follows. The categorical ingredients are standard: the canonical presheaf is an image factorization, the coarsest quotient is an intersection of kernels with its universal property, and the terminal observed system is the limit of the observation

presheaf, as the labels on Theorems 11.1, 11.3, and 13.4 record. What is proposed is a unifying representation framework in which these ingredients are applied jointly to context, observation, disclosure, authority, and transitions, together with comparative results that the ingredients alone do not supply. Three of these are the center of the essay. First, regional and accumulative information flow arise as dual product constructions on one decorated tree, and the refinement categories they generate are isomorphic only when the tree is a chain (Proposition 13.7). Second, observational equivalence and behavioral equivalence are kept visibly apart: Example 13.6 exhibits two systems with one observational abstraction, differing in a bounded counter, so that a translation exists in one direction and none exists in the other, and the example is the model case of the distinction between what an abstraction retains and what a stronger equivalence demands. Third, the representation theorem for tree systems (Theorem 17.2) is stated for specified observational models over a common tree and makes no claim of equivalence between filesystem and lexical-scope semantics. The framework is compared with institutions and with scope graphs in Appendix E.

The word context is used in at least two technical senses that must not be conflated. In the theory of programs a context is a term with a hole, and two terms are contextually equivalent when no context distinguishes their behavior [25]. Here a context is a position or region of a system from which observations are made and operations are attempted, such as a scope, a directory, a process, or a room. The two senses meet in Section 16, where quantification over all program contexts is seen as quantification over a family of observation contexts in the second sense.

Part I (Sections 2 to 4) develops the mathematics of context: contexts as objects of a preorder or category, the independence of knowledge, visibility, and authority, and the separation of navigation from disclosure. Part II (Sections 5 to 8) examines four families of examples: filesystems and shells, lexical scope, access control, and interactive worlds. Part III (Sections 9 to 11) introduces the presheaf of observations, the local-to-global reconstruction problem, and the quotient of states by observational equivalence. Part IV (Sections 12 to 16) defines the category of contextual representation systems and studies translation, covering common abstractions, the distinction among surjectivity, fullness, faithfulness, and equivalence, translation among mathematical languages, and universality. Part V (Sections 17 to 20) assembles the general construction and its representation theorems, analyzes preservation and loss, examines the limits of hierarchical representation, and concludes.

2 Contexts as Mathematical Objects

Definition 2.1 (Context system). A *context system* is a triple $\mathcal{C} = (C, \preceq, T)$ in which C is a set of contexts, $\preceq \subseteq C \times C$ is a relation of nesting, read $c \preceq d$ as “ c is nested in d ”, and $T \subseteq C \times L \times C$ is a set of labeled navigation transitions over a set L of labels.

Nesting is naturally reflexive and transitive: every context lies within itself, and a context within a context within a third lies within the third. Reflexivity and transitivity make $\preceq$ a preorder. Antisymmetry is a further demand that fails in practice whenever two distinct

contexts are mutually nested, as happens when two names denote one directory through a bind mount, or when a module is entered through two aliases. Retaining the preorder keeps both names, while passing to the poset quotient identifies them, which is harmless for questions about observations and harmful for questions about naming. The relation supplied by a presentation is usually not transitive but is the generating relation of its reflexive transitive closure, as with the parent-child edges of a directory tree, whose closure is the ancestor relation. The distinction between a generating relation and the preorder it generates returns in Section 19.

Several shapes of nesting occur in practice and need to be distinguished. A nested hierarchy is one in which each context has a single immediate container. A graph with multiple parents arises from hard links, multiple inheritance, and shared modules. A graph with cycles arises from mutually recursive modules and from portals in interactive worlds; in the reflexive transitive closure a cycle collapses its vertices into one equivalence class, so a preorder cannot distinguish a cycle from a set of aliases. Overlapping contexts, such as two windows or two permission regions that intersect without either containing the other, are most naturally modeled as subsets of a set of locations ordered by inclusion, with intersection as meet whenever it exists, a point developed in Section 9. The hierarchical case admits a crisp characterization.

Proposition 2.2 (Rooted trees as posets; ESTABLISHED). *A finite poset $(C, \preceq)$ is a rooted tree under the ancestor relation, with the root as greatest element, if and only if it has a greatest element and, for every $c \in C$, the set $\uparrow c = \{d : c \preceq d\}$ is a chain.*

Proof. In a rooted tree with $c \preceq d$ meaning that d lies on the path from c to the root, $\uparrow c$ is that path, which is a chain, and the root is greatest. Conversely, given a greatest element $\top$ and chains $\uparrow c$, the set $\uparrow c \setminus \{c\}$ is a finite chain, nonempty when $c \neq \top$ because it contains $\top$, so it has a least element, which is declared the parent of c . The resulting parent function defines a rooted tree whose ancestor relation is $\preceq$. $\square$

The consequence used repeatedly is that in a rooted tree the ancestors of any node are linearly ordered, so a node with two incomparable ancestors cannot occur in any tree.

Proposition 2.3 (Categories from contexts; ESTABLISHED). *(a) A preorder $(C, \preceq)$ is a thin category $\mathcal{C}_{\preceq}$ with a unique arrow $c \rightarrow d$ exactly when $c \preceq d$. Functors between such categories are monotone maps, and the category is skeletal exactly when $\preceq$ is antisymmetric.*
(b) A directed graph G generates a free category $\text{Path}(G)$ whose arrows are finite directed paths, the empty path included, and a full functor $\text{Path}(G) \rightarrow \text{Reach}(G)$ onto the preorder of reachability, which is the identity on objects. This functor is faithful if and only if G is acyclic and has at most one directed path between any two vertices.

Proof. Part (a) is the standard identification of a preorder with a category in which composition is transitivity and identities are reflexivity. For (b), $\text{Path}(G)$ is full over $\text{Reach}(G)$ because every reachable pair is joined by some path. The functor is faithful exactly when each hom-set of $\text{Path}(G)$ has at most one element, which fails when two distinct paths join the same vertices,

and fails at a vertex lying on a directed cycle, where the empty path and the cycle are distinct arrows $v \rightarrow v$. $\square$

The distinction in part (b) between routes and reachability is the first instance of a pattern that organizes the essay: a representation can retain the arrow or only the fact that an arrow exists, and the two give different theories. A graph with vertices h, a, b, r and edges $h \rightarrow a$, $h \rightarrow b$, $a \rightarrow r$, $b \rightarrow r$ has two arrows $h \rightarrow r$ in $\text{Path}(G)$ and one in $\text{Reach}(G)$; whether a system treats these as the same position depends on whether the route taken is part of the state. Section 5 shows that the logical working directory of a shell retains the route and the physical working directory retains only the endpoint.

Two orders on the same set of contexts will be in play and must not be confused. The nesting preorder $\preceq$ records containment, while Section 9 introduces a refinement order recording which context observes more. For a filesystem viewed through subtrees the two coincide, and for lexical scope they are opposite, a fact whose consequences occupy Sections 13 and 17.

3 Scope, Visibility, and Authority

Fix a set I of informational items, such as bindings, directory entries, or facts about a world, and a set Σ of operations. To each context c three subsets are assigned. The set $K(c) \subseteq I$ is the information available to an observer located at c . The set $V(c) \subseteq K(c)$ is the information exposed by a particular interface at c , subject to the design discipline that an interface cannot present what is unavailable. The set $A(c) \subseteq \Sigma$ is the set of operations admissible at c . Knowledge, presentation, and authority are independent in the precise sense that none determines the others. A hidden file lies in $K(c)$ but not $V(c)$ until a disclosure is performed; a read-only file lies in $V(c)$ while the write operation on it lies outside $A(c)$; an operation may be admissible at a context whose interface no longer displays the object it acts upon. The interesting question is how these assignments vary along the nesting relation, and the answer is that nesting alone constrains none of them.

Proposition 3.1 (No forced monotonicity; PROVED HERE). *Let $c \prec d$ be strictly nested contexts, and let I and Σ be sufficiently large sets of information items and actions. For each of K , V , and A and for each of the relations strict inclusion, reverse strict inclusion, equality, and incomparability, there exist assignments of subsets of I or Σ to c and d for which the values at c and d stand in that relation. For V the assignments are required to satisfy $V(e) \subseteq K(e)$ at each context e , so the relation between $V(c)$ and $V(d)$ is realizable for each of the four relations but is not independent of the relation between $K(c)$ and $K(d)$; for example, $K(c) = \emptyset$ forces $V(c) = \emptyset$. Moreover, on a three-element chain $c_1 \prec c_2 \prec c_3$, each assignment can be neither monotone nor antitone.*

Proof. For the first claim, choose subsets of I or Σ realizing the desired relation; for V , choose $V(c)$ and $V(d)$ among the subsets of $K(c)$ and $K(d)$ respectively, which is possible for each relation because $V = K$ and $V = \emptyset$ are both permitted. For the second claim take $I = \{1, 2, 3, 4\}$, $\Sigma = \{r, w, x\}$, and the following assignment.

	c_1	c_2	c_3
K	$\{1, 2\}$	$\{1\}$	$\{1, 3\}$
V	$\{1\}$	$\{1\}$	$\{3\}$
A	$\{r\}$	$\{w\}$	$\{r, w, x\}$

Here $K(c_1) \supsetneq K(c_2) \subsetneq K(c_3)$, while $V(c_2)$ and $V(c_3)$ are incomparable and $A(c_1)$ and $A(c_2)$ are incomparable. Hence none of the three assignments is monotone or antitone along the chain, and $V \subseteq K$ holds throughout. $\square$

The finite counterexample is not an artifact. Lexical scope makes the information available to an inner block grow with depth, a sandbox or a changed root directory makes it shrink, a setuid program or a privilege-granting command makes authority grow on entry, and attenuation of a capability makes it shrink. Hierarchical context inclusion therefore entails no monotonicity of visibility or authority, and any claim that deeper contexts see less, or can do less, or see more, or can do more, is a property of a specific discipline rather than of nesting.

The next definition fixes the primitive construction on which the remainder of the essay is built. It follows a deliberate convention about the direction of arrows: an arrow $u : c \rightarrow d$ means that c is a restriction of d , so that d is the observationally richer of the two. This convention is chosen so that observations restrict along arrows in the manner of a presheaf, and it is independent of the nesting order, which may agree with it or oppose it.

Definition 3.2 (Contextual representation system). A *contextual representation system* is a quintuple

$$\mathfrak{R} = (\mathcal{C}, X, \Omega, \mathcal{A}, \delta)$$

together with an alphabet Σ of actions, consisting of the following.

- (1) A small category $\mathcal{C}$ of contexts. An arrow $u : c \rightarrow d$ is a *refinement arrow*: c is a restriction of d .
- (2) A set X of underlying states.
- (3) A presheaf $\Omega : \mathcal{C}^{\text{op}} \rightarrow \mathbf{Set}$ of observation spaces, with restriction maps $\Omega(u) : \Omega(d) \rightarrow \Omega(c)$ for $u : c \rightarrow d$, together with observation maps $o_c : X \rightarrow \Omega(c)$.
- (4) A context-indexed assignment of admissible actions $\mathcal{A} : \text{Ob}(\mathcal{C}) \rightarrow \mathcal{P}(\Sigma)$.
- (5) A partial transition function $\delta : X \times \Sigma \rightharpoonup X$.

The observation maps must satisfy *observation coherence*: for every arrow $u : c \rightarrow d$,

$$o_c = \Omega(u) \circ o_d.$$

Coherence is a modeling assumption, not a law of operating systems, and several real disciplines violate it as stated; Section 9 shows that the flat visible environment of a language with shadowing is one, and repairs it. The definition requires neither that authority nor that action sets increase with observational richness. Movement of an observer from one context

to another is not an arrow of $\mathcal{C}$; it is a transition of the system, represented by δ over a state space that records position, or by the explicit configuration spaces of Section 4. This keeps the arrows of $\mathcal{C}$ for the single purpose of carrying the coherence of observations.

Theorem 3.3 (Observational refinement; PROVED HERE). *For every context c let $x \sim_c y$ if and only if $o_c(x) = o_c(y)$. Then for every arrow $u : c \rightarrow d$,*

$$\sim_d \subseteq \sim_c.$$

Proof. If $x \sim_d y$ then $o_d(x) = o_d(y)$, and applying $\Omega(u)$ and using coherence gives $o_c(x) = \Omega(u)o_d(x) = \Omega(u)o_d(y) = o_c(y)$. $\square$

This is the first substantive consequence of the definition: richer observations distinguish at least as many states as their restrictions. It is a statement about observation only, and it carries no implication for authority.

Proposition 3.4 (Independence of authority; PROVED HERE). *Observational refinement implies neither $\mathcal{A}(c) \subseteq \mathcal{A}(d)$ nor $\mathcal{A}(d) \subseteq \mathcal{A}(c)$ for an arrow $u : c \rightarrow d$.*

Proof. Let $\mathcal{C}$ have two objects c, d and one non-identity arrow $u : c \rightarrow d$. Let $X = \{0, 1\}$, $\Omega(d) = \{0, 1\}$, $\Omega(c) = \{*\}$, with $\Omega(u)$ the constant map, $o_d = \text{id}$, and o_c constant. Coherence holds, and $\sim_d$ is the diagonal while $\sim_c$ identifies both states, consistent with Theorem 3.3. Let $\Sigma = \{a, b\}$ with $\mathcal{A}(c) = \{a\}$ and $\mathcal{A}(d) = \{b\}$. The two action sets are disjoint and nonempty, so neither inclusion holds. $\square$

Proposition 3.4 separates visibility from permission formally, and Section 7 shows that the order for which authority is monotone is a different order again, the order of delegation.

4 Navigation and Disclosure as Distinct Operators

An observer's situation is a pair consisting of a position and a disclosure state, the latter recording display options such as whether hidden entries are shown or whether filename extensions are displayed. Let S be a set of disclosure states and let $\text{Conf} = C \times S$ be the set of configurations. A navigation transition $t = (c, \ell, c') \in T$ induces a partial map

$$N_t : \text{Conf} \rightharpoonup \text{Conf}, \quad (c, s) \mapsto (c', \rho_t(s)),$$

defined on $\{c\} \times S$, where $\rho_t : S \rightarrow S$ records the policy by which display settings are carried across the transition. A family $f = (f_c)_{c \in C}$ of partial maps $f_c : S \rightharpoonup S$ induces a disclosure operator

$$D_f : \text{Conf} \rightharpoonup \text{Conf}, \quad (c, s) \mapsto (c, f_c(s)),$$

which changes the disclosure state within the same context. Both are typed as partial maps on Conf , so that the composites $N_t D_f$ and $D_f N_t$ are meaningful and the question of commutation is well posed; without this typing the equation $ND = DN$ is shorthand and nothing more.

Proposition 4.1 (Commutation criterion; PROVED HERE). *Let $t = (c, \ell, c')$. If all maps involved are total on S , the composites $N_t D_f$ and $D_f N_t$ agree on $\{c\} \times S$ if and only if*

$$\rho_t \circ f_c = f_{c'} \circ \rho_t \quad \text{on } S.$$

In particular:

- (i) *if $\rho_t = \text{id}_S$, navigation and disclosure commute exactly when $f_c = f_{c'}$;*
- (ii) *if ρ_t is constant with value s_0 , they commute exactly when $f_{c'}(s_0) = s_0$;*
- (iii) *if f_c is defined at s and $f_{c'}$ is undefined at $\rho_t(s)$, then $N_t D_f$ is defined at (c, s) while $D_f N_t$ is not, so the composites differ as partial maps.*

Proof. Computing, $N_t D_f(c, s) = (c', \rho_t(f_c(s)))$ and $D_f N_t(c, s) = (c', f_{c'}(\rho_t(s)))$, which are equal for all s exactly when the displayed identity holds. Case (i) substitutes $\rho_t = \text{id}$. Case (ii) gives $s_0 = f_{c'}(s_0)$. Case (iii) is read off from the definitions of the composites as partial maps. $\square$

The identity $\rho_t f_c = f_{c'} \rho_t$ is an intertwining condition, the statement that the policy ρ_t is equivariant with respect to the disclosure operators on the two sides of the transition. Its failure has everyday instances. In a file manager that stores a “show hidden entries” setting per folder, $S = \{0, 1\}^C$, $\rho_t = \text{id}$, and f_c toggles the coordinate of c ; then $f_c \neq f_{c'}$ and the order of toggling and navigating matters, since toggling and then navigating carries the toggled state to the new folder’s configuration while navigating and then toggling acts on the new folder’s own coordinate. When the setting is global, $S = \{0, 1\}$ and f_c is the same toggle everywhere, and navigation and disclosure commute. When entering a folder resets the display to a default s_0 at which the toggle is not a fixed point, case (ii) shows that disclosure performed before navigation is always lost, whereas disclosure performed after navigation is not. A virtual folder with no notion of hidden entries realizes case (iii).

Disclosure at a fixed context is governed by a refinement relation among observation maps, which is the same structure as Theorem 3.3 at a single context.

Definition 4.2 (Disclosure refinement). At a fixed context c , let $o_c^{(0)} : X \rightarrow Y_0$ and $o_c^{(1)} : X \rightarrow Y_1$ be two observation maps. Observation 1 *refines* observation 0 if there exists $r : Y_1 \rightarrow Y_0$ with $o_c^{(0)} = r \circ o_c^{(1)}$.

Theorem 4.3 (Disclosure refinement; PROVED HERE). *If observation 1 refines observation 0 then $\ker(o_c^{(1)}) \subseteq \ker(o_c^{(0)})$. The inclusion is strict exactly when the richer observation separates at least one pair of states that the coarser one identifies.*

Proof. If $o_c^{(1)}(x) = o_c^{(1)}(y)$ then applying r gives $o_c^{(0)}(x) = o_c^{(0)}(y)$. Strictness is the statement that the two kernels differ, which is the existence of a pair in $\ker(o_c^{(0)}) \setminus \ker(o_c^{(1)})$. $\square$

This formalizes the difference between an ordinary directory listing and one that also reveals dotfiles, and equally the difference between a listing that shows base names and one that shows names with extensions: in each case the second observation determines the first

by discarding detail, and the discarded detail is exactly what the strict inclusion of kernels measures. Disclosure moves along refinement within a context, whereas navigation moves between contexts, and Proposition 4.1 identifies when the two can be reordered.

Part II

Hierarchical and Computational Representations

5 Filesystems and Shell Environments

A filesystem state can be modeled as a structure $F = (N, N_{\text{dir}}, r, \text{ent}, \text{mode})$ in which N is a set of nodes, $N_{\text{dir}} \subseteq N$ the directories, $r \in N_{\text{dir}}$ the root, $\text{ent} : N_{\text{dir}} \times \text{Names} \rightarrow N$ the directory-entry function, and $\text{mode} : N \rightarrow \text{Modes}$ an assignment of permission bits. Symbolic links are nodes that carry a target path, and the entries `.` and `..` are ordinary entries of each directory, the latter naming the physical parent. Path resolution is the partial function $\text{res}_F : N_{\text{dir}} \times \text{Paths} \rightarrow N$ obtained by iterating ent along the components of a path, expanding symbolic links when they are met. In the special case where the graph of ent , omitting `.`, `..`, and links, is a rooted tree, the filesystem is a hierarchy in the sense of Proposition 2.2. Hard links to files make the structure a directed acyclic graph in which a file has several parents, so that the path to a file is no longer unique, and symbolic links add edges that need not respect the tree at all.

The working directory of a shell illustrates the difference between a route and an endpoint recorded in Proposition 2.3. A POSIX shell maintains a logical working directory as a path string, and the built-in that changes directory behaves differently under the logical option and the physical option when symbolic links are present [32]. If `link` in `/home/u` is a symbolic link to `/var/data`, then changing into `link` and then into `..` returns to `/home/u` under logical resolution and moves to `/var` under physical resolution, because the physical parent of `/var/data` is `/var`. The parent of a context is therefore not a function of the node alone once links are present: the logical working directory retains an arrow of the path category, and the physical working directory retains only a point of the reachability preorder. A correct model of the context in which a relative path is interpreted must say which of the two the shell uses, and the logical choice is the unfolding of the graph into paths that Section 19 analyzes.

A process adds a second layer that is distinct from the filesystem. A process state includes a working directory, an environment mapping variable names to strings, a file-creation mask, credentials, and open file descriptors. The process tree is generated by spawning: a child created by `fork` receives a copy of its parent's state at the moment of creation, and a subshell is such a child. Inheritance is therefore a snapshot and not a live reference, since later changes to the parent's environment do not reach an existing child and changes made by the child vanish when it exits. This contrasts with a closure in a lexically scoped language, which

captures the location of a binding and so remains coupled to later updates, as Section 6 shows. A third layer is the command interpreter, whose resolution of a command name consults an ordered list: aliases, then shell functions, then built-ins, then the directories named in the search path in order, and the first match wins. This ordered list has the form of a scope chain, but the order is fixed by the interpreter and not by nesting.

Changing the working directory, entering a subshell, and altering file permissions share contextual features, for each changes what some later operation will see, yet they are mathematically distinct. To make this precise, let the configuration of the system be $F \times \text{Proc}^+$, a filesystem state together with a nonempty tree of processes with one designated current process. Let cd_d change the working directory of the current process, defined when the process holds search permission on d ; let sub create a child copy of the current process and make it current; let exit discard the current process and return to its parent; and let $\text{chmod}_{d,m}$ set the mode of d to m , defined for the owner.

- Proposition 5.1** (Distinctness of cd , sub , and chmod ; PROVED HERE). *(a) The three operations modify different components of the configuration $F \times \text{Proc}^+$: cd_d changes only the working-directory coordinate of the designated process, sub changes only the process tree and the designation of the current process, and $\text{chmod}_{d,m}$ changes only the mode coordinate of F . Their domains of definition nevertheless read other components, since cd_d consults the modes in F and the credentials of the current process, and $\text{chmod}_{d,m}$ consults the ownership recorded in F against those credentials.*
- (b) They have different persistence. For every finite sequence w of environment-modifying operations and working-directory changes performed between sub and the matching exit , the composite $\text{exit} \circ w \circ \text{sub}$ restores the parent's coordinates to their former values, whereas cd_d alters the working directory of the current process and $\text{chmod}_{d,m}$ alters the permissions seen by every process.*
- (c) They do not commute. If an unprivileged process owns a directory d on which it holds search permission, then the sequence cd_d followed by $\text{chmod}_{d,000}$ is defined, while the sequence $\text{chmod}_{d,000}$ followed by cd_d is undefined.*

Proof. Parts (a) and (b) are read off from the definitions of the operations as maps on $F \times \text{Proc}^+$: a child receives a copy of the parent's state, so operations performed on the child leave the parent's coordinates untouched, and discarding the child at exit restores the parent's private process coordinates. Effects on shared components, such as modifications of F made by the child, persist after the child exits, and (b) is claimed only for the parent's private coordinates. For (c), after cd_d the process holds d as working directory, and the owner may then clear all permission bits on d . In the reverse order, cd_d requires search permission on the target directory under standard semantics, which the cleared mode denies, so the second step is undefined. $\square$

The three operations therefore cannot be modeled by one kind of arrow in a single category of contexts. Changing directory is a change of position, entering a subshell is a scoped extension of the process component whose effects are discarded on exit, and changing permissions is

a change of the authority assignment of the shared component. The table summarizes the profile.

Operation	Component changed	Persistence	Seen by
cd_d	current process, working directory	until next change	this process
sub with environment changes	process tree, child coordinates	until exit	the child and its descendants
$\text{chmod}_{d,m}$	filesystem, mode of d	until next change	all processes

6 Lexical Scope and Abstract Syntax Trees

A program is parsed into an abstract syntax tree, and the binding forms of the language, such as `let`, `lambda`, function definitions, loops, and comprehensions, introduce scopes within it. Each occurrence of an identifier is a reference to be resolved to a declaration. The tree of syntax is not by itself the structure that resolution consults, because resolution follows binding and not syntactic adjacency, and a more faithful structure is the scope graph [29].

Definition 6.1 (Scope graph and resolution policy). A *scope graph* is a directed graph $G = (V, E)$ of scopes in which every vertex v carries a partial binding function $b_v : \text{Names} \rightarrow \text{Val}$. A *resolution policy* assigns to each pair (v, n) of a scope and a name a finite ordered sequence $\varrho(v, n) = (v_1, \dots, v_k)$ of eligible scopes, each reachable from v along E , with $v_1 = v$ permitted. The resolution of n at v is

$$\text{res}(v, n) = b_{v_i}(n) \quad \text{for the least } i \text{ with } n \in \text{dom } b_{v_i},$$

and is undefined when no such i exists.

Theorem 6.2 (Determinacy of resolution; PROVED HERE). *For a resolution policy of the form in Definition 6.1, res is a partial function from pairs (v, n) to values.*

Proof. A finite ordered sequence has a unique first position at which a defined binding occurs whenever any such position exists, so the value selected is determined by (v, n) . $\square$

The theorem applies to a precisely specified resolution mechanism and not automatically to every language, since languages with dynamic evaluation of names or with ambiguous imports may not supply a finite ordered sequence. Python is a useful test case because its rule is often summarized as local, enclosing, global, built-in, and that summary hides features that distinguish the policy ϱ from simple traversal of ancestors [33]. At compile time every name occurring in a function body is classified. A name that is bound anywhere in the body, by assignment, import, definition, or loop variable, is local unless declared otherwise, and this classification covers binding occurrences that follow the use. For a local name, $\varrho(v, n) = (v)$, so that if the binding has not yet been made when the lookup executes, resolution fails and does not continue outward. A name that is used but not bound locally is free, and its eligible

scopes are the enclosing function scopes from the innermost outward, then the module scope, then the built-in scope, with enclosing class scopes omitted. The following fragments exhibit both departures from ancestor traversal.

```
x = 1
def f():
    print(x)          # UnboundLocalError: x is local to f
    x = 2

x = 'global'
class A:
    x = 'class'
    def m(self):
        return x      # evaluates to 'global', not 'class'
```

The policy therefore depends on the name through a classification of the entire body, and it skips class scopes. Two further features concern what a resolved binding is. A closure captures the location of a binding and not its value, so that the expression `[lambda: i for i in range(3)]` produces three functions that all return the final value of `i`, which is the contrast with the snapshot behavior of process environments noted in Section 5. An import statement binds a module or copies a reference at import time, a snapshot again. Finally, lexical and dynamic scope differ in which parent relation resolution follows. For lexical scope the relevant relation is the static nesting of definitions, and for dynamic scope it is the chain of callers at run time. In the program that defines `x = 1`, a function `f` returning `x`, and a function `g` that binds `x = 2` and calls `f`, the call to `g` yields 1 under lexical scope and 2 under dynamic scope. Two distinct parent relations are thus attached to the same set of activation contexts, and conflating them is the source of many incorrect claims that scope is a tree.

Proposition 6.3 (Shadowing is non-injective; PROVED HERE). *Let environments be partial maps $E : \text{Names} \rightarrow \text{Val}$ and let extension be $E \mapsto E[n \mapsto v]$. This operation is not injective. Consequently the observable lookup behavior of an extended environment does not determine the environment that was extended.*

Proof. Let E_1 and E_2 be distinct environments that differ only at the name n . Then $E_1[n \mapsto v] = E_2[n \mapsto v]$. $\square$

Shadowing also governs how the information visible at a scope varies with nesting. Define the information available at a scope v as the partial map $K(v) : n \mapsto (\text{declaring scope}, \text{res}(v, n))$, so that $K(v)$ records which declaration a name resolves to.

Proposition 6.4 (Visibility under nesting; PROVED HERE). *Let G be a rooted scope tree with the policy $\varrho(v, n) = (v, \text{par}(v), \text{par}^2(v), \dots, \text{root})$ for every name.*

- (i) *If w is nested in v then $\text{dom } K(v) \subseteq \text{dom } K(w)$: an inner scope resolves at least the names that its container resolves.*

- (ii) *The graph of $K(v)$ is contained in the graph of $K(w)$ if and only if no name in $\text{dom } K(v)$ is rebound in a scope strictly between v and w or in w itself.*
- (iii) *Under the Python policy, which omits class scopes from the eligible scopes of nested functions, statement (i) can fail: if a class body v binds a name n that is bound nowhere else, a method w nested in v does not resolve n .*

Proof. For (i) the chain of ancestors of v is a suffix of the chain of ancestors of w , so any scope that supplies a binding at v is eligible at w . For (ii), if some name $n \in \text{dom } K(v)$ is rebound at an intermediate scope or at w then $K(w)(n)$ names a different declaring scope, so the pair belonging to $K(v)$ is absent from $K(w)$; conversely if no rebinding occurs then every pair of $K(v)$ persists. For (iii) the class scope is not eligible from the method, which is the second fragment above. $\square$

Visibility is therefore monotone in nesting at the level of domains in the idealized chain policy, monotone at the level of declarations only in the absence of shadowing, and not monotone even at the level of domains in a real language. This is the concrete content behind the abstract Proposition 3.1.

Proposition 6.5 (Scope graphs strictly generalize scope trees; PROVED HERE). *Every rooted scope tree determines a scope graph, with an edge from each scope to its parent, whose reachability preorder is the ancestor relation. Not every scope graph admits a tree representation that preserves all edges and reachability relations bijectively.*

Proof. The first claim is the identification of the parent relation with a graph. For the second, consider the directed acyclic graph with vertices d, b, c, a and edges $d \rightarrow b$, $d \rightarrow c$, $b \rightarrow a$, $c \rightarrow a$, in which d has two incomparable parents b and c . In a rooted tree the ancestors of any node form a chain by Proposition 2.2, so b and c cannot both be ancestors of d . A tree representation therefore either discards one of the edges $d \rightarrow b$ and $d \rightarrow c$, or duplicates d into two nodes, one below each parent, after which the correspondence on nodes is no longer bijective. $\square$

The diamond is not a contrived case: a class with two base classes in a language with multiple inheritance realizes it, and so does a module reached through two import paths. This supplies a formal reason not to assume that every contextual system is literally a nested tree, a point taken up again in Section 19.

7 Access Control and Contextual Authority

The access-matrix model of protection associates to each pair of a principal and an object a set of rights [19]. A capability is an unforgeable reference that carries rights and can be passed from one holder to another [10]. Discretionary access control, as in the permission bits of a Unix filesystem, attaches rights to objects and grants them according to the identity and group of the requester. In a capability discipline there is no ambient authority, so that

authority arises only from references actually held [7]. Execution environments inherit part of their authority from their creators, but inheritance is mediated by explicit mechanisms such as dropping privileges, changing identity on execution of a privileged program, and passing descriptors.

The natural question is whether authorization is a functorial assignment to contexts, that is, whether a context-indexed set of admissible operations is monotone with respect to the relevant order, so that $\mathcal{A} : (C, \leq) \rightarrow (\mathcal{P}(\Sigma), \subseteq)$ is a functor. The answer depends entirely on which order is meant, because three distinct orders are attached to the same execution contexts. The first is containment, the nesting of directories or scopes. The second is spawn, the process tree in which each process is created by another. The third is delegation, in which $c \sqsubseteq d$ when the authority of c is obtained by passing a subset of the authority of d . Authority is monotone with respect to delegation by construction, and the content of the next proposition is that it need not be monotone with respect to either of the others.

Proposition 7.1 (Authority and context orders; PROVED HERE). *There exist a finite set of contexts, three orders on it for containment, spawn, and delegation, and an assignment $\mathcal{A}$ such that $\mathcal{A}$ is monotone with respect to delegation and not monotone with respect to containment or spawn. In particular authorization need not decrease with depth in a hierarchy.*

Proof. Let the contexts be u , a user shell with working directory `/home`; l , a sandboxed child with working directory `/home/sandbox`; and s , a program with elevated authority started by u with working directory `/home/bin`. Containment gives $l \preceq u$ and $s \preceq u$. Spawn gives l and s as children of u . Delegation gives $l \sqsubseteq u$ only, since s holds authority that u did not pass to it. Let $\mathcal{A}(u) = \{\text{read}, \text{write}\}$, $\mathcal{A}(l) = \{\text{read}\}$, and $\mathcal{A}(s) = \{\text{read}, \text{write}, \text{admin}\}$. Then $\mathcal{A}(l) \subseteq \mathcal{A}(u)$, consistent with delegation, while $s \preceq u$ and s is a child of u with $\mathcal{A}(s) \not\subseteq \mathcal{A}(u)$, so monotonicity fails for containment and for spawn. $\square$

A second distinction concerns path-based and handle-based checks. Reaching a file by name requires search permission on every directory along the path, so the right to reach decreases, in the sense of a conjunction, along the path. After a file has been opened, however, the descriptor is a capability, and later access is checked against the object and the descriptor's rights and not against the ancestors of the original path. The same object thus has authority that is decreasing in depth for name resolution and independent of depth for held references. A limit on any attempt to translate authority structure wholesale is the undecidability of the safety problem for the general access-matrix model, which asks whether a right can leak to a principal that lacks it [17]; Section 18 returns to what translations can and cannot preserve about prohibitions.

8 Interactive Worlds as Transition Systems

An interactive environment is represented as a labeled transition system $\mathcal{G} = (S, \Sigma, \longrightarrow)$ with $\longrightarrow \subseteq S \times \Sigma \times S$, in which a state records everything that can matter to the next step and an action label records the input or event. Levels, subworlds, menus, portals, and nested

interaction states are interpreted as contextual partitions: a surjection $q : S \rightarrow C$ assigns each state to the context it occupies, and the *context graph* Γ_q has an edge $c \xrightarrow{a} c'$ whenever some transition $s \xrightarrow{a} s'$ has $q(s) = c$ and $q(s') = c'$. Platform games with a map of worlds and levels, games that navigate a graph of cells joined by portals, and strategy games with stacked interface screens are instances of this architecture, and they differ in which kind of nesting they use. Spatial nesting is containment of geometric regions, as of rooms in a floor of a building. Logical nesting is a stack of modes, as of a menu opened over a level. State-dependent accessibility is the presence of an edge only in some states, as of a door that opens only when a key is held.

Proposition 8.1 (Context graphs over-approximate; ESTABLISHED). *The map q sends every run of $\mathcal{G}$ to a run of Γ_q , so q is a simulation of $\mathcal{G}$ by Γ_q . The converse can fail: with initial state s_0 and initial context $q(s_0)$, the graph Γ_q may contain runs from $q(s_0)$ that no run of $\mathcal{G}$ from s_0 projects to.*

Proof. The first claim is the definition of Γ_q . For the failure, let $S = \{A, B\} \times \{0, 1\}$ with the context map $q(r, k) = r$, initial state $s_0 = (A, 0)$, an action **take** from $(A, 0)$ to $(A, 1)$, and an action **go** from $(A, 1)$ to $(B, 1)$ only. Then Γ_q has the edge $A \xrightarrow{\text{go}} B$, so the run consisting of **go** alone is a run of Γ_q from the initial context A , but from $s_0 = (A, 0)$ no run of $\mathcal{G}$ begins with **go**. $\square$

State-dependent accessibility is therefore not recoverable from the unguarded context graph, and a faithful representation must either attach guards, which are predicates on S , to the edges of Γ_q , or enlarge the context set to include the relevant part of the state.

Logical nesting raises a different difficulty, because the target of the action that closes a menu is the context from which the menu was opened and not a function of the current context.

Proposition 8.2 (Modal return is not an edge of the context graph; PROVED HERE). *Let a system have contexts $c_1, \dots, c_n$ with $n \geq 2$, from each of which an action **open** leads to a context m , and from m an action **close** returns to the context from which m was entered. Suppose c_1 and c_2 are distinguishable, in that some action is enabled at one and not the other. Then no quotient of the system that identifies configurations with the same current context is bisimilar to it. If menus may open menus without bound, the system is not finite-state in the context component.*

Proof. In the quotient, m has **close**-edges to every c_i , so the quotient admits the run $c_1, \text{open}, m, \text{close}, c_2$, which the system does not. Bisimilarity would require that the configuration $(m, \text{origin } c_1)$ be matched to m and that the quotient move $m \xrightarrow{\text{close}} c_2$ be matched by a system move from $(m, \text{origin } c_1)$, whose only **close**-successor is c_1 ; since c_1 and c_2 are distinguishable, no matching exists. For unbounded nesting, the return target is the top of a stack of unbounded depth, which is the structure of a pushdown system and not of a finite graph [39]. $\square$

When the nesting depth is bounded, the stack can be folded into the context by taking contexts to be bounded sequences, which restores a finite graph at the price of enlarging the context set. This is the first instance of a general fact developed in Section 19, that a graph can be replaced by a hierarchy only by unfolding, and that the unfolding preserves behavior without preserving the identification of repeated contexts.

Part III

Presheaves and Local Information

9 Presheaf Semantics for Contextual Observation

A presheaf $F : \mathcal{C}^{\text{op}} \rightarrow \mathbf{Set}$ assigns to each context c a set $F(c)$ of admissible local descriptions, observations, or states, and to each arrow $u : c \rightarrow d$ a restriction map $F(u) : F(d) \rightarrow F(c)$ satisfying $F(\text{id}_c) = \text{id}_{F(c)}$ and $F(vu) = F(u) \circ F(v)$ for composable $u : c \rightarrow d$ and $v : d \rightarrow e$. Restriction is a map of descriptions of one and the same situation taken at a finer or coarser resolution, and it goes from the richer context to the poorer one. The observation presheaf Ω of Definition 3.2 is of this kind, and coherence says that the observation maps $o_c : X \rightarrow \Omega(c)$ form a natural transformation from the constant presheaf on X to Ω . Two families of examples show how different the underlying systems can be while sharing this form.

Example 9.1 (Region systems). Let L be a set of locations, let $(V_\ell)_{\ell \in L}$ be sets of local values, and let $X = \prod_{\ell \in L} V_\ell$. Let $\mathcal{C}$ be a family of subsets of L ordered by inclusion, with an arrow $U \rightarrow W$ whenever $U \subseteq W$, and put $\Omega(U) = \prod_{\ell \in U} V_\ell$, with $\Omega(U \rightarrow W)$ the projection and $o_U : X \rightarrow \Omega(U)$ the projection. Coherence is the composition of projections. Arrows point from smaller regions to larger ones, and larger regions are richer. A filesystem viewed through subtrees, where $\Omega(d)$ is the content of the subtree rooted at d , and a game world viewed through its rooms are both of this form, with locations the nodes or cells and the local values their attributes.

Example 9.2 (Accumulation systems). Let T be a finite rooted tree ordered so that $q \preceq p$ when q is nested in p , let $(L(p))_{p \in T}$ be sets of frames, and let $X = \prod_{p \in T} L(p)$. Let $\mathcal{C} = T^{\text{op}}$, with an arrow $p \rightarrow q$ whenever q is nested in p , so that arrows point from outer scopes to inner ones, and put $\Omega(q) = \prod_{p \succeq q} L(p)$, the product over the chain of ancestors of q including q . For the arrow $p \rightarrow q$, the map $\Omega(q) \rightarrow \Omega(p)$ is the projection onto the ancestors of p , and $o_q(x) = (x_p)_{p \succeq q}$. Coherence holds. Inner scopes are richer: the observation at q includes the frames of every enclosing scope.

In Example 9.1 the refinement arrows agree with nesting and in Example 9.2 they oppose it, which is the structural difference between a system of sections over regions and a system of environments built by accumulation. The second example uses frames, the full ancestor chain, and not the flat map of visible names, and this is forced by coherence.

Proposition 9.3 (Flat environments violate coherence; PROVED HERE). *In an accumulation system with an outer scope p and an inner scope q nested in p , each with frame set $\{0, 1\}$ for a variable of the same name, define the flat observation by $o_p^{\text{flat}}(x) = x_p$ and $o_q^{\text{flat}}(x) = x_q$, so that the inner binding shadows the outer one. Then there is no restriction map $\Omega(q) \rightarrow \Omega(p)$ making the arrow $p \rightarrow q$ coherent, whereas the frame observations $o_p(x) = x_p$ and $o_q(x) = (x_p, x_q)$ are coherent.*

Proof. Coherence for the arrow $p \rightarrow q$ would force $\sim_q \subseteq \sim_p$ by Theorem 3.3. The flat relation $\sim_q$ identifies the states $(x_p, x_q) = (0, 0)$ and $(1, 0)$ because both have $x_q = 0$, while $\sim_p$ separates them, so the inclusion fails. For the frame observations, $\Omega(q) = \{0, 1\}^2 \rightarrow \Omega(p) = \{0, 1\}$ is the first projection, and $o_p = \Omega(u) \circ o_q$. $\square$

This repairs the situation noted after Definition 3.2: the flat visible environment of a language with shadowing is not a coherent observation, and the coherent one retains shadowed bindings. It also shows why Proposition 6.3 is the algebraic face of an observational fact, since non-injectivity of extension is exactly the loss of information that coherence forbids.

The mathematical restriction of information must be distinguished from the act of navigating into a subcontext. A restriction map relates two descriptions of the same state at different resolutions, whereas navigation moves an observer. In an accumulation system, moving inward increases the information available to the observer, while the restriction map from the inner description to the outer one discards information. In a region system, moving into a subregion decreases the information available, and the restriction map to the subregion also decreases it. The presheaf encodes neither movement, only the consistency of descriptions across resolutions, and movement belongs to the transition structure.

Remark 9.4 (Class scopes separate nesting from refinement). In Python the body of a class statement is a scope, but names bound in it are not visible from the function bodies defined inside it [33]. Taking the enclosing function as a parent, the class scope and each method scope are therefore siblings for the purpose of name resolution, even though textually the methods are nested in the class. The nesting preorder of the source text and the refinement order of the observation maps then differ, which is a concrete instance of the distinction between the two orders drawn in Section 2.

10 Local Consistency and Global Reconstruction

The passage from local descriptions to a global one requires a notion of covering. A *sieve* on a context c is a set S of arrows with codomain c closed under precomposition. A *Grothendieck topology* J assigns to each context a collection $J(c)$ of covering sieves satisfying the usual axioms [22]. For a presheaf P and a sieve S on c , a *matching family* is a family $(s_u)_{u \in S}$ with $s_u \in P(\text{dom } u)$ and $P(v)(s_u) = s_{uv}$ whenever $uv \in S$, and P is a *sheaf for S* if every matching family has a unique *amalgamation*, an element $s \in P(c)$ with $P(u)(s) = s_u$ for all $u \in S$. Uniqueness is the *separation* condition, and existence is the *gluing* condition. When contexts are regions with intersections and S is generated by a family $U_i \subseteq W$, a family (s_i) extends

to a matching family exactly when s_i and s_j agree on $U_i \cap U_j$, which is the familiar condition of agreement on overlaps. The following is the standard criterion, stated for orientation.

Proposition 10.1 (Gluing criterion; ESTABLISHED). *For a Grothendieck topology J and a covering sieve S on c , compatible local data for S admit a unique global datum exactly when the presheaf is a sheaf for S .*

This is the definition of the sheaf condition and is not a new theorem. The contribution lies in determining which contextual systems satisfy it and in producing explicit failures. For a contextual representation system, the relevant presheaf is the canonical presheaf Q of observational distinctions constructed in Section 11, and the question has two parts that the following definition separates.

Definition 10.2 (Failure modes of reconstruction). Let S be a covering sieve on c in a contextual representation system.

- (F1) *Incompatible information*: a family of local descriptions, not necessarily arising from any state, that disagree on an overlap, so that it is not a matching family.
- (F2) *Missing information*: a matching family defined only on a proper subsieve of S , which may extend to S in several ways or in none.
- (F3) *Obstruction*: a matching family for S that is not realized by any state.
- (F4) *Ambiguity*: two distinct global descriptions at c , in particular two states distinguished at c , that determine the same matching family for S .

The four cases correspond to different defects. Incompatible information is a defect of the data and rules out any global state. Missing information is a defect of coverage and is repaired by supplying more data. Obstruction is a defect of the system, in that locally consistent data cannot be globally realized because some constraint is invisible to every local context. Ambiguity is the dual defect, in that some global distinction is invisible to every local context. Incompatibility is immediate from coherence.

Proposition 10.3 (Disagreement excludes realization; PROVED HERE). *If contexts c_1, c_2 have arrows $c_{12} \rightarrow c_1$ and $c_{12} \rightarrow c_2$ and local descriptions $s_1 \in \Omega(c_1)$ and $s_2 \in \Omega(c_2)$ have different restrictions to c_{12} , then no state x satisfies both $o_{c_1}(x) = s_1$ and $o_{c_2}(x) = s_2$.*

Proof. By coherence both $o_{c_1}(x)$ and $o_{c_2}(x)$ restrict to $o_{c_{12}}(x)$. □

Obstruction and ambiguity are not excluded by coherence and are exhibited by finite examples. The first example is an obstruction.

Example 10.4 (Locally acyclic dependencies; obstruction). Let $E = \{e_1, e_2, e_3\}$ be the edges of a triangle on the vertices 1, 2, 3, with $e_1 = \{1, 2\}$, $e_2 = \{2, 3\}$, and $e_3 = \{3, 1\}$. Let $\mathcal{C}$ be the poset of subsets of E , with an arrow $U \rightarrow W$ when $U \subseteq W$, and let a sieve cover W when the union of its domains is W , which is a Grothendieck topology on the subsets of a set. Let X be the set of acyclic orientations of the triangle, which has six elements. Write

an orientation of e_i as $+$ when it points in the direction $1 \rightarrow 2 \rightarrow 3 \rightarrow 1$ and $-$ otherwise, let $\Omega(U) = \{+, -\}^U$, and let o_U restrict an orientation to the edges in U . This is a region system. For the three singletons, $Q(\{e_i\}) = \{+, -\}$ because each orientation of a single edge is realized, and $Q(\emptyset)$ is a point. The sieve S on E generated by the singletons has pairwise empty overlaps, so every triple of local orientations is a matching family, and $\text{Match}_S(Q)$ has eight elements. The two families $(+, +, +)$ and $(-, -, -)$ are the directed cycles and are not realized by any acyclic orientation. Hence Q satisfies separation but not gluing for S .

Read in terms of programs, the three edges are dependencies between pairs of modules. Each dependency considered alone is consistent, and each pair of dependencies is consistent, yet the global invariant that the dependency relation be acyclic is a nonlocal constraint that no local observation exposes. The second example is an ambiguity.

Example 10.5 (Global-only observable; ambiguity). Let $L = \{a, b\}$ and $\mathcal{C}$ the poset of subsets of L , with the same covering convention, and let $X = \{0, 1\}^3$ with coordinates (x_a, x_b, p) . Let $\Omega(U) = \{0, 1\}^U$ for $U \subsetneq L$ and $\Omega(L) = \{0, 1\}^3$, with the evident projections and o_U the corresponding projections of X . Then $\sim_L$ is the diagonal while the sieve S on L generated by $\{a\}$ and $\{b\}$ has $\bigcap \sim_{\{a\}} \cap \sim_{\{b\}}$ identifying any two states that agree on x_a and x_b . The states $(0, 0, 0)$ and $(0, 0, 1)$ are distinguished at L and have the same local classes at both members of S , so separation fails: the coordinate p is an observable that is visible only at the top context.

Example 10.4 exhibits a family of locally compatible descriptions that cannot be realized by any single underlying state, and Example 10.5 exhibits states that admit more than one global realization of the same family. These are the two failures that the gluing condition is designed to detect. Sheaf-theoretic formulations of this phenomenon in physics, in which empirical data are consistent on overlapping contexts yet admit no global assignment, are developed in the study of contextuality [2], and for abelian-valued presheaves the obstruction to gluing is measured by Čech cohomology [22].

11 Hidden Information and Observational Equivalence

Given a contextual representation system, Theorem 3.3 defines at each context the equivalence relation $x \sim_c y$ when $o_c(x) = o_c(y)$ and shows that these relations are nested along arrows. Their quotients assemble into a presheaf that is canonical in a precise sense. Distinct states may become indistinguishable under a restricted observation interface, and moving to a richer context or increasing disclosure, as in Theorem 4.3, refines the relation and separates more states.

Theorem 11.1 (Canonical presheaf of observational distinctions; ESTABLISHED, applied here). *For a contextual representation system, let $Q(c) = X/\sim_c$ and, for an arrow $u : c \rightarrow d$, let $Q(u) : Q(d) \rightarrow Q(c)$ be $[x]_d \mapsto [x]_c$.*

- (a) *Q is a well-defined presheaf, and the quotient maps $p_c : X \rightarrow Q(c)$ form a natural transformation $p : \Delta X \Rightarrow Q$ with every component surjective.*

- (b) The observations factor as $o_c = \iota_c \circ p_c$ with $\iota_c : Q(c) \rightarrow \Omega(c)$ the injection $[x]_c \mapsto o_c(x)$, and $\iota : Q \Rightarrow \Omega$ is natural. This is the epi-mono factorization of $o : \Delta X \Rightarrow \Omega$ in the category of presheaves.
- (c) (Universal property.) Let P be a presheaf, $\pi : \Delta X \Rightarrow P$ a natural transformation with surjective components, and $\omega : P \Rightarrow \Omega$ a natural transformation with $o = \omega\pi$. Then there is a unique natural transformation $\theta : P \Rightarrow Q$ with $\theta\pi = p$, and $\iota\theta = \omega$. In this sense Q is the coarsest presheaf quotient of ΔX through which the observations factor.

Proof. (a) Theorem 3.3 gives $\sim_d \subseteq \sim_c$, so $Q(u)$ is well defined. Functoriality is $[x]_e \mapsto [x]_d \mapsto [x]_c$, equal to $Q(vu)$ for $u : c \rightarrow d$ and $v : d \rightarrow e$, and identities are clear. Naturality of p is the definition of $Q(u)$. (b) Injectivity of ι_c is the definition of $\sim_c$, and $\Omega(u)\iota_d[x]_d = \Omega(u)o_d(x) = o_c(x) = \iota_c Q(u)[x]_d$ gives naturality. Epi and mono in the presheaf category are detected componentwise. (c) If $\pi_c(x) = \pi_c(y)$ then $o_c(x) = \omega_c \pi_c(x) = \omega_c \pi_c(y) = o_c(y)$, so $x \sim_c y$, and $\theta_c(\pi_c(x)) = [x]_c$ is well defined and unique because π_c is surjective. Naturality of θ follows from naturality of π and p and surjectivity of π_d , since $\theta_c P(u)\pi_d(x) = \theta_c \pi_c(x) = [x]_c = Q(u)\theta_d \pi_d(x)$. Finally $\iota\theta\pi = \iota p = o = \omega\pi$ and surjectivity of π give $\iota\theta = \omega$. $\square$

This is the bridge between context-relative observations and presheaf semantics: a system of coherent observations generates a presheaf of observational equivalence classes, and the presheaf is canonical. The factorization is an instance of image factorization in a presheaf topos and is not itself surprising. What it gives is a standard against which any proposed abstraction can be measured, as in Section 13.

Proposition 11.2 (Global indistinguishability; PROVED HERE). *Two states are indistinguishable at every context exactly when $(x, y) \in \bigcap_{c \in \text{Ob}(\mathcal{C})} \sim_c$. The combined observation family separates states if and only if this intersection is the diagonal relation on X .*

Proof. Both statements are immediate from the definition of $\sim_c$ as the kernel of o_c . $\square$

A canonical common abstraction depends on which contexts one chooses to observe through, and the next theorem identifies the abstraction relative to an arbitrary family of contexts and proves that it is universal. For a family $\mathcal{U} \subseteq \text{Ob}(\mathcal{C})$ put $\sim_{\mathcal{U}} = \bigcap_{c \in \mathcal{U}} \sim_c$, $X_{\mathcal{U}} = X / \sim_{\mathcal{U}}$, and let $q_{\mathcal{U}} : X \rightarrow X_{\mathcal{U}}$ be the quotient. A surjection $f : X \rightarrow Z$ is $\mathcal{U}$ -sound if for every $c \in \mathcal{U}$ the observation o_c factors through f , that is, $o_c = g_c \circ f$ for some $g_c : Z \rightarrow \Omega(c)$. A family $\mathcal{U}$ is *generating* if every context c admits an arrow $c \rightarrow d$ with $d \in \mathcal{U}$.

The canonical presheaf above and the coarsest quotient below are standard constructions, an image factorization in a presheaf category and an intersection of kernels, applied to observation maps. Their content is the identification of the factorization with the observational structure of a contextual system, and the quotient they produce is canonical relative to the chosen observations, not independent of them: enlarging or shrinking the family $\mathcal{U}$ changes the quotient, and Appendix C computes the change on a four-node example.

Theorem 11.3 (Coarsest quotient compatible with a family of contexts; PROVED HERE). (a) $q_{\mathcal{U}}$ is $\mathcal{U}$ -sound.

- (b) Every $\mathcal{U}$ -sound surjection $f : X \rightarrow Z$ factors uniquely as $q_{\mathcal{U}} = k \circ f$ for a map $k : Z \rightarrow X_{\mathcal{U}}$. Equivalently, $q_{\mathcal{U}}$ is the terminal object of the category whose objects are $\mathcal{U}$ -sound surjections out of X and whose morphisms are maps commuting with them.
- (c) If $\mathcal{U}$ is generating then $\sim_{\mathcal{U}} = \bigcap_{c \in \text{Ob}(\mathcal{C})} \sim_c$, so the quotient is the global indistinguishability quotient and is determined by the members of $\mathcal{U}$ alone.

Proof. (a) For $c \in \mathcal{U}$, $\sim_{\mathcal{U}} \subseteq \sim_c$, so o_c is constant on classes of $\sim_{\mathcal{U}}$ and induces $g_c : X_{\mathcal{U}} \rightarrow \Omega(c)$ with $o_c = g_c q_{\mathcal{U}}$. (b) If f is $\mathcal{U}$ -sound then $\ker f \subseteq \ker o_c = \sim_c$ for each $c \in \mathcal{U}$, so $\ker f \subseteq \sim_{\mathcal{U}} = \ker q_{\mathcal{U}}$. Hence $k(f(x)) = q_{\mathcal{U}}(x)$ is well defined, and unique because f is surjective. (c) For c with an arrow $c \rightarrow d$, $d \in \mathcal{U}$, Theorem 3.3 gives $\sim_d \subseteq \sim_c$, so $\sim_{\mathcal{U}} \subseteq \sim_d \subseteq \sim_c$. $\square$

Part (c) shows that the canonical global abstraction is governed by the richest contexts, and these differ by system. For a filesystem viewed through subtrees the richest context is the root, so the global quotient is determined by the observation at the root. For a lexical accumulation system the richest contexts are the innermost scopes, so two states indistinguishable at every innermost scope are indistinguishable everywhere. The dependence on the orientation of arrows already appears at the level of what the canonical abstraction is built from.

The relation to Section 10 is recorded as a criterion for Q itself.

Proposition 11.4 (Gluing criterion for Q ; PROVED HERE). *Let S be a covering sieve on c generated by arrows $u_i : c_i \rightarrow c$. Then Q is a sheaf for S if and only if*

- (i) $\sim_c = \bigcap_i \sim_{c_i}$, and
- (ii) every matching family of Q for S is of the form $([x]_{c_i})_i$ for some state $x \in X$.

Condition (i) fails in Example 10.5 and condition (ii) fails in Example 10.4.

Proof. Elements $[x]_c$ and $[y]_c$ have the same restrictions to every c_i exactly when $(x, y) \in \bigcap_i \sim_{c_i}$, so separation is the inclusion $\bigcap_i \sim_{c_i} \subseteq \sim_c$, and the reverse inclusion holds by Theorem 3.3. An amalgamation of a matching family is a class $[x]_c$ with $[x]_{c_i}$ equal to the prescribed class for every i , which exists exactly when some state x realizes the family. The two examples were computed in Section 10. $\square$

A system with dynamics has a further equivalence beyond static observation: two states may be indistinguishable now and still be separated by some sequence of actions. Call an equivalence relation R on X a δ -congruence if $x R y$ implies, for every $a \in \Sigma$, that $\delta(x, a)$ is defined exactly when $\delta(y, a)$ is, and that then $\delta(x, a) R \delta(y, a)$.

Proposition 11.5 (Behavioral minimization; ESTABLISHED). *Among δ -congruences contained in $\bigcap_c \sim_c$ there is a largest, written $\approx$. The quotient $\mathfrak{R}/\approx = (\mathcal{C}, X/\approx, \Omega, \mathcal{A}, \delta_{\approx})$ with $\delta_{\approx}([x], a) = [\delta(x, a)]$ is a contextual representation system, and $\approx$ is the identity relation on it.*

Proof. The transitive closure of the union of all δ -congruences contained in $\bigcap_c \sim_c$ is an equivalence relation contained in $\bigcap_c \sim_c$, because the latter is transitive, and it is a δ -congruence by applying the defining condition along a chain. It is therefore the largest. The operation $\delta_{\approx}$

is well defined by the congruence property, the observations descend because $\approx$ is contained in each $\sim_c$, and maximality gives that no nontrivial congruence remains on the quotient. $\square$

This is Moore-machine minimization with the profile of observations at all contexts as output [24, 37], and it identifies the canonical abstraction of a system with dynamics as its behaviorally reduced quotient.

Part IV

Functorial Translation

12 Categories of Representational Systems

A comparison of two contextual representation systems needs a notion of structure-preserving map, and the choice of which components to preserve, and in what sense, determines everything that follows. Let $\mathfrak{R} = (\mathcal{C}, X, \Omega, \mathcal{A}, \delta)$ with alphabet Σ and $\mathfrak{S} = (\mathcal{C}', X', \Omega', \mathcal{A}', \delta')$ with alphabet Σ' be systems, with observation maps o and o' .

Definition 12.1 (Translation). A *translation* $\Phi = (F, f, \eta, \alpha) : \mathfrak{R} \rightarrow \mathfrak{S}$ consists of

- (T1) a functor $F : \mathcal{C} \rightarrow \mathcal{C}'$;
- (T2) a map $f : X \rightarrow X'$ of states;
- (T3) maps $\eta_c : \Omega(c) \rightarrow \Omega'(Fc)$ for every context c , natural in the sense that $\eta_c \circ \Omega(u) = \Omega'(Fu) \circ \eta_d$ for every arrow $u : c \rightarrow d$;
- (T4) a map $\alpha : \Sigma \rightarrow \Sigma'$ of actions,

subject to the following conditions.

- (O) *Observation preservation*: $o'_{Fc} \circ f = \eta_c \circ o_c$ for every context c .
- (A) *Lax authority preservation*: $\alpha(\mathcal{A}(c)) \subseteq \mathcal{A}'(Fc)$ for every context c .
- (D) *Simulation of dynamics*: if $\delta(x, a) = y$ then $\delta'(f(x), \alpha(a)) = f(y)$.

A translation is *observational* if only (T1) to (T3) and (O) are required. It is *authority-reflecting* if $\mathcal{A}(c) = \alpha^{-1}(\mathcal{A}'(Fc))$, and *strong* if in addition $\delta(x, a)$ is defined exactly when $\delta'(f(x), \alpha(a))$ is.

The minimal requirement for the mapping on contexts to qualify as a functor is that it preserve identities and composition, which is not automatic for an arbitrary assignment of arrows. Where the category of contexts is generated by a graph of navigation edges, freeness gives a convenient way to meet this requirement, since a map of generating edges to arrows of the target extends uniquely to a functor on the path category. The conditions (A) and (D) are *lax*: they require inclusion and implication in one direction, not equality, and the strong variants are the strict counterparts. The observation condition (O) is strict.

Proposition 12.2 (The category **CRS**; PROVED HERE). *Translations compose by $(F', f', \eta', \alpha') \circ (F, f, \eta, \alpha) = (F'F, f'f, \eta'_{F(-)} \circ \eta_{(-)}, \alpha'\alpha)$, with identities $(\text{id}, \text{id}, \text{id}, \text{id})$, and this makes systems and translations into a category **CRS**. The authority-reflecting and strong translations are closed under composition.*

Proof. For naturality of the composite $\eta'_{Fc}\eta_c$, take $u : c \rightarrow d$ and compute $\eta'_{Fc}\eta_c\Omega(u) = \eta'_{Fc}\Omega'(Fu)\eta_d = \Omega''(F'Fu)\eta'_{Fd}\eta_d$, using naturality of η and then of η' at the arrow Fu . For (O), $o'_{F'Fc}f'f = \eta'_{Fc}o'_{Fc}f = \eta'_{Fc}\eta_c o_c$. For (A), $\alpha'\alpha(\mathcal{A}(c)) \subseteq \alpha'(\mathcal{A}'(Fc)) \subseteq \mathcal{A}''(F'Fc)$. For (D), apply the hypothesis twice. Identities satisfy all conditions, and associativity is inherited from composition of maps and functors. For the authority-reflecting case, $(\alpha'\alpha)^{-1}(\mathcal{A}''(F'Fc)) = \alpha^{-1}(\alpha'^{-1}(\mathcal{A}''(F'Fc))) = \alpha^{-1}(\mathcal{A}'(Fc)) = \mathcal{A}(c)$, and the strong case is the same argument applied to definedness. $\square$

The first preservation result is that every translation preserves observational indistinguishability, and the converse needs more.

Theorem 12.3 (Translation preserves source indistinguishability; PROVED HERE). *For every translation $\Phi : \mathfrak{R} \rightarrow \mathfrak{S}$, every context c of $\mathfrak{R}$, and all states x, y of $\mathfrak{R}$,*

$$x \sim_c y \implies f(x) \sim'_{Fc} f(y).$$

Proof. If $o_c(x) = o_c(y)$ then $o'_{Fc}f(x) = \eta_c o_c(x) = \eta_c o_c(y) = o'_{Fc}f(y)$ by (O). $\square$

Proposition 12.4 (Reflection requires additional structure; PROVED HERE). *The converse of Theorem 12.3 does not hold in general. It holds at c for every translation in which η_c is injective on $o_c(X)$, and conversely if the converse holds at c then η_c is injective on $o_c(X)$.*

Proof. For the failure take $X = X' = \{0, 1\}$, $f = \text{id}$, $\Omega(c) = \{0, 1\}$ with $o_c = \text{id}$, and $\Omega'(c) = \{*\}$ with η_c constant. Then $f(0) \sim'_c f(1)$ while $0 \not\sim_c 1$. If η_c is injective on $o_c(X)$ and $f(x) \sim'_{Fc} f(y)$, then $\eta_c o_c(x) = o'_{Fc}f(x) = o'_{Fc}f(y) = \eta_c o_c(y)$, so $o_c(x) = o_c(y)$. Conversely, if the converse holds and $\eta_c(o_c(x)) = \eta_c(o_c(y))$, then $f(x) \sim'_{Fc} f(y)$ by (O), hence $x \sim_c y$ and $o_c(x) = o_c(y)$. $\square$

Preservation and reflection of distinctions are thus mathematically separate. A translation that merges the observation space of a context collapses distinctions, and no property of the target can recover them.

13 Common Abstractions and Forgetful Functors

Two systems have a *common abstraction* when both translate into a third system. Forgetting components yields natural families of such abstractions. The reduct that forgets $\mathcal{A}$ and δ sends a system to its observational structure $(\mathcal{C}, X, \Omega, o)$; the reduct that forgets δ alone retains authority; and each is a functor from **CRS**, since the conditions defining translation restrict to the retained components. A translation can be classified by what it covers and what it retains.

Definition 13.1 (Kinds of representation). A translation $\Phi = (F, f, \eta, \alpha)$ is *state-surjective* if f is surjective, *context-surjective* if F is surjective on objects, and *observation-reflecting* if η_c is injective on $o_c(X)$ for every c , equivalently by Proposition 12.4 if $x \sim_c y \Leftrightarrow f(x) \sim'_{F_c} f(y)$ for all c . It is a *surjective representation* if it is state-surjective and context-surjective, a *faithful representation* if it is moreover observation-reflecting, and an *isomorphism of systems* if it has an inverse translation in **CRS**.

The first result states what a common surjective abstraction does and does not yield. It is valid and elementary, and its modesty is the point: an abstraction retains only the distinctions that its observation maps carry.

Theorem 13.2 (Common abstraction without equivalence; PROVED HERE). *Let $\mathfrak{R}_1$ and $\mathfrak{R}_2$ be contextual representation systems admitting observational translations $\Phi_i = (F_i, f_i, \eta^i) : \mathfrak{R}_i \rightarrow \mathfrak{S}$ into a common system $\mathfrak{S}$, and suppose the state maps are surjective, $f_i : X_i \twoheadrightarrow X_{\mathfrak{S}}$. Then:*

- (i) *every state of $\mathfrak{S}$ has at least one representative in each X_i ;*
- (ii) *for every context c , $x \sim_c^i y$ implies $f_i(x) \sim_{F_i c}^{\mathfrak{S}} f_i(y)$, and target observations pull back consistently: if $s \in X_{\mathfrak{S}}$ is represented by $x_1 \in X_1$ and $x_2 \in X_2$, and $F_1 c_1 = F_2 c_2 = e$, then $\eta_{c_1}^1 o_{c_1}^1(x_1) = \eta_{c_2}^2 o_{c_2}^2(x_2) = o_e^{\mathfrak{S}}(s)$;*
- (iii) *$\mathfrak{R}_1$ and $\mathfrak{R}_2$ need not be isomorphic, need not have state sets of the same cardinality, and need not have isomorphic context categories.*

Proof. Part (i) is surjectivity and part (ii) is Theorem 12.3 together with condition (O). For (iii), let $\mathfrak{S}$ be the system with one context, one state, a trivial observation space, and no actions. Let $\mathfrak{R}_1$ have one state and $\mathfrak{R}_2$ two states, each with one context, trivial observations, and no actions, so that both map onto $\mathfrak{S}$ by constant maps, yet $|X_1| \neq |X_2|$ rules out an isomorphism. For context categories, let $\mathcal{C}_1$ be the terminal category and $\mathcal{C}_2$ the one-object category whose endomorphisms are $\{1, e\}$ with $e^2 = e$, each with a one-point state set, trivial observations, and no actions. Both admit functors to the terminal category, and $\mathcal{C}_1 \not\cong \mathcal{C}_2$. $\square$

Common abstraction is cheap in both directions. There is a terminal system, so that every system translates to it and every pair of systems has a common abstraction that retains nothing, and there is a disjoint-union system that retains everything and identifies nothing.

Proposition 13.3 (Trivial common abstractions; PROVED HERE). **CRS** *has a terminal object $\mathfrak{T}$, with one context, one state, a one-point observation space, one action σ admissible at the context, and $\delta(*, \sigma) = *$. Consequently any two systems have a common state-surjective and context-surjective abstraction.*

Proof. For any $\mathfrak{R}$ there is exactly one functor $\mathcal{C} \rightarrow 1$, one map $X \rightarrow \{*\}$, one family of maps $\Omega(c) \rightarrow \{*\}$, and one map $\Sigma \rightarrow \{\sigma\}$. Condition (O) holds because the target has one point, (A) holds because $\{\sigma\} \subseteq \mathcal{A}'(*)$, and (D) holds because $\delta'(*, \sigma) = * = f(y)$. Uniqueness of each component gives uniqueness of the translation. $\square$

A significant common abstraction therefore has to be singled out by what it retains. If a family of observations is fixed, there is a canonical choice.

Theorem 13.4 (Terminal observed system; ESTABLISHED, applied here). *Fix a small category $\mathcal{C}$ and a presheaf $\Omega : \mathcal{C}^{\text{op}} \rightarrow \mathbf{Set}$. An Ω -observed set is a pair (X, o) with $o : \Delta X \Rightarrow \Omega$ natural, and a morphism $h : (X, o) \rightarrow (X', o')$ is a map with $o'h = o$. Let $\Gamma(\Omega) = \lim \Omega$ be the set of global sections, that is, families $s = (s_c)_c \in \prod_c \Omega(c)$ with $\Omega(u)(s_d) = s_c$ for every arrow $u : c \rightarrow d$, and let $\text{ev}_c : \Gamma(\Omega) \rightarrow \Omega(c)$ be the projections. Then $(\Gamma(\Omega), \text{ev})$ is the terminal Ω -observed set, and the unique morphism from (X, o) is the profile map*

$$\pi : X \rightarrow \Gamma(\Omega), \quad \pi(x) = (o_c(x))_{c \in \text{Ob}(\mathcal{C})}.$$

Proof. By coherence $\Omega(u)(o_d(x)) = o_c(x)$ for $u : c \rightarrow d$, so $\pi(x)$ is a global section, and $\text{ev}_c \pi = o_c$. If h is any morphism, $\text{ev}_c h(x) = o_c(x)$ for every c , which determines $h(x)$ as the family $\pi(x)$. $\square$

Corollary 13.5 (What the canonical abstraction retains and forgets; PROVED HERE). *For an Ω -observed set with profile map π :*

- (a) π is injective exactly when the combined observations separate states, and $X / \bigcap_c \sim_c \cong \text{im } \pi$, which identifies the global indistinguishability quotient of Proposition 11.2 with the set of realized profiles;
- (b) π is surjective exactly when every global section of Ω is realized by a state;
- (c) two Ω -observed sets (X_1, o^1) and (X_2, o^2) have the same realized profiles, $\text{im } \pi_1 = \text{im } \pi_2$, exactly when they have the same canonical abstraction, and they are isomorphic as Ω -observed sets exactly when $|\pi_1^{-1}(s)| = |\pi_2^{-1}(s)|$ for every $s \in \Gamma(\Omega)$.

Proof. Parts (a) and (b) restate Proposition 11.2 and surjectivity. For (c), an isomorphism h satisfies $\pi_2 h = \pi_1$, so it carries each fiber of π_1 bijectively to the fiber of π_2 over the same profile, and conversely fiberwise bijections assemble to an isomorphism. $\square$

The corollary identifies precisely what the canonical abstraction forgets about the observed set, namely the multiplicity of microstates over each realized profile, and what remains to be compared separately, namely $\mathcal{A}$ and δ . Part (b) is the global form of the existence half of the gluing condition and part (a) the global form of separation. The difference between a common abstraction and an equivalence is made concrete by the following pair of systems.

Example 13.6 (Same observational abstraction, no translation back; PROVED HERE). Let $\mathcal{C} = \{c \rightarrow d\}$ with $\Omega(d) = \{0, 1\}^2$, $\Omega(c) = \{0, 1\}$, and $\Omega(c \rightarrow d)$ the first projection. Let $\Sigma = \{\text{flip}\}$, $\mathcal{A}(c) = \emptyset$, and $\mathcal{A}(d) = \{\text{flip}\}$. Let $\mathfrak{R}_1$ have $X_1 = \{0, 1\}^2$, $o_d = \text{id}$, o_c the first coordinate, and $\delta_1(x, \text{flip}) = (1 - x_1, x_2)$. Let $\mathfrak{R}_2$ have $X_2 = \{0, 1\}^2 \times \{0, 1, 2\}$ with $o_d(x, k) = x$, $o_c(x, k) = x_1$, and $\delta_2((x, k), \text{flip}) = ((1 - x_1, x_2), k + 1)$ when $k < 2$ and undefined when $k = 2$. The projection $\mathfrak{R}_2 \rightarrow \mathfrak{R}_1$ forgetting the counter, with $F = \text{id}$, $\eta = \text{id}$, and $\alpha = \text{id}$, is a translation that is state-surjective and observation-reflecting, and both systems realize every profile, so their canonical abstractions coincide. There is, however, no

translation $\mathfrak{R}_1 \rightarrow \mathfrak{R}_2$ at all. If h were the state map, the run $x_0 \rightarrow x_1 \rightarrow x_0$ of $\mathfrak{R}_1$ with $x_0 = (0, 0)$ and $x_1 = (1, 0)$ would be sent by (D) to a run $h(x_0) \rightarrow h(x_1) \rightarrow h(x_0)$ of $\mathfrak{R}_2$, but δ_2 increases the counter at every step, which is impossible. Hence $\mathfrak{R}_1$ and $\mathfrak{R}_2$ are not isomorphic in **CRS**.

The example separates the observational abstraction from the dynamics: $\mathfrak{R}_2$ can be flipped at most twice, and the abstraction does not see this. This is the sense in which an abstraction established relative to observations can hold while the systems differ in operational behavior.

A second family of comparisons connects the structural examples of Part II. Let T be a finite rooted tree, ordered by nesting with the root greatest, and let $L = (L(p))_{p \in T}$ be a family of local sets. The pair (T, L) is a *decorated rooted tree*, and it generates two systems on the common state set $X = \prod_p L(p)$, with $\mathcal{A} = \emptyset$ and no transitions. The *region system* $\mathfrak{R}^\downarrow(T, L)$ has $\mathcal{C}^\downarrow = (T, \preceq)$ with an arrow $q \rightarrow p$ when $q \preceq p$, observation space $\Omega^\downarrow(p) = \prod_{q \preceq p} L(q)$ over the nodes nested in p , and o_p the projection. This is the filesystem viewed through subtrees, and a world viewed through its regions. The *accumulation system* $\mathfrak{R}^\uparrow(T, L)$ has $\mathcal{C}^\uparrow = (T, \preceq)^{\text{op}}$, observation space $\Omega^\uparrow(q) = \prod_{p \succeq q} L(p)$ over the ancestors of q , and o_q the projection, as in Example 9.2. This is lexical scope. Both are coherent.

Proposition 13.7 (Orientation duality; PROVED HERE). *Let (T, L) be a decorated rooted tree with all $L(p)$ nonempty.*

- (a) *The nesting categories of the filesystem model and of the lexical-scope model, each generated by the parent relation, are isomorphic whenever the parent trees are isomorphic.*
- (b) *The profile maps of $\mathfrak{R}^\downarrow(T, L)$ and $\mathfrak{R}^\uparrow(T, L)$ are both bijections onto their global sections, so both systems have the identity as global indistinguishability relation, and $\Gamma(\Omega^\downarrow) \cong X \cong \Gamma(\Omega^\uparrow)$.*
- (c) *The refinement categories $\mathcal{C}^\downarrow = (T, \preceq)$ and $\mathcal{C}^\uparrow = (T, \preceq)^{\text{op}}$ are isomorphic if and only if T is a chain. In particular, if T has a node with two children, no isomorphism of context categories carries the region system to the accumulation system.*

Proof. (a) is the statement that the parent relation has the same orientation in both models. For (b), a global section of $\Omega^\downarrow$ is determined by its component at the root, which lies in $\prod_q L(q) = X$ and restricts to every other component, so $\Gamma(\Omega^\downarrow) \cong X$ and π is the identity under this identification. For $\Omega^\uparrow$, every node is an ancestor of some leaf, and a compatible family of ancestor chains over all leaves agrees on shared ancestors, hence defines a unique element of $\prod_p L(p)$, so $\Gamma(\Omega^\uparrow) \cong X$ and π is again the identity. For (c), the poset $(T, \preceq)$ has exactly one maximal element, the root, and as many minimal elements as leaves. A chain is isomorphic to its dual. If T has a node with two children then it has at least two leaves, so its dual has at least two maximal elements and one minimal element, and an order isomorphism must preserve the number of maximal elements. $\square$

This clarifies a tempting formulation: a filesystem and a lexical scope structure are the same tree. They share the parent tree and the state space, and they share their global

observational abstraction, but the information flows in opposite directions along it. The two systems differ in the lattice of local partitions: the region system has $\sim_p^\downarrow$ identifying states that agree on the subtree of p , and the accumulation system has $\sim_q^\uparrow$ identifying states that agree on the ancestor chain of q , and these two families are not related by any isomorphism of the indexing categories when T branches. The common abstraction is the decorated tree with its undirected rooted structure, and the orientation of information flow is an additional structure that the abstraction does not share. Nor does a shared decorated tree establish equivalence of file operations and name-resolution semantics, which differ in $\mathcal{A}$ and δ . Moreover, a filesystem and a game world arise from the same decorated tree. A hub with two rooms, each carrying one local bit, is the region system of a directory with two subdirectories each carrying one file attribute, and the two have identical observational structure and different dynamics.

	Filesystem on (T, L)	Game world on (T, L)
contexts	directories	hub and rooms
local content $L(p)$	entry attributes	room object states
observation at p	listing of subtree	view of region
actions Σ	cd, chmod, rm	move, take, toggle
authority $\mathcal{A}$	mode bits and ownership	key and door conditions

14 Surjectivity, Fullness, Faithfulness, and Equivalence

The vocabulary of functors separates several properties that informal talk about mappings between systems runs together. A functor $F : \mathcal{C} \rightarrow \mathcal{D}$ is *full* if it is surjective on each hom-set, *faithful* if it is injective on each hom-set, and *essentially surjective* if every object of $\mathcal{D}$ is isomorphic to some $F(c)$. Surjectivity on objects is stronger than essential surjectivity and independent of the other two.

Theorem 14.1 (Equivalence criterion; ESTABLISHED). *A functor $F : \mathcal{C} \rightarrow \mathcal{D}$ is an equivalence of categories if and only if it is full, faithful, and essentially surjective [21].*

This is the standard benchmark for the stronger equivalence claim. Everything weaker is a statement about a subset of these properties, and the next result shows that the weakest, surjectivity on objects, implies none of the others.

Proposition 14.2 (Surjectivity does not imply equivalence; PROVED HERE). *A functor that is surjective on objects need not be full or faithful, and need not be an equivalence.*

Proof. Let $\mathcal{D}_2$ be the discrete category with two objects a, b and let 1 be the terminal category. The unique functor $\mathcal{D}_2 \rightarrow 1$ is surjective on objects but not full, since $\text{Hom}(a, b) = \emptyset$ while $\text{Hom}(Fa, Fb) = \{\text{id}\}$. It is not an equivalence, because equivalent categories have the same number of isomorphism classes of objects. For faithfulness, let M be the one-object category with endomorphisms $\{1, e\}$ with $e^2 = e$. The functor $M \rightarrow 1$ is surjective on objects and full, and not faithful because 1 and e have the same image. It is not an equivalence because corresponding hom-sets of equivalent categories have the same cardinality. $\square$

Proposition 14.3 (No unrestricted faithful collapse; PROVED HERE). *A functor that identifies two distinct parallel arrows is not faithful. In particular the reflection $\text{Path}(G) \rightarrow \text{Reach}(G)$ of Proposition 2.3 is not faithful whenever G has two distinct routes between the same vertices, so no faithful translation of contexts can collapse a distinction between routes.*

Proof. Faithfulness is injectivity on every hom-set, so two distinct arrows with the same source and target and the same image violate it. $\square$

Rigidity of posets places a sharp limit on what categorical equivalence can establish for context hierarchies.

Proposition 14.4 (Equivalence of posets is isomorphism; PROVED HERE). *A functor between posets, regarded as categories, is an equivalence if and only if it is an order isomorphism.*

Proof. An order isomorphism is an equivalence. Conversely, an equivalence is full and faithful, so it reflects the order, $F(p) \leq F(q) \Rightarrow p \leq q$. If $F(p) = F(q)$ then $p \leq q$ and $q \leq p$, so $p = q$ by antisymmetry and F is injective. In a poset the only isomorphisms are identities, so essential surjectivity is surjectivity on objects, and F is a bijective order-reflecting monotone map. $\square$

Corollary 14.5 (Morita rigidity for posets; ESTABLISHED). *Two posets have equivalent categories of presheaves if and only if they are isomorphic.*

Proof. Two small categories have equivalent presheaf categories exactly when their Cauchy completions are equivalent [6]. A poset has no idempotents other than identities, so its Cauchy completion is equivalent to itself, and Proposition 14.4 gives isomorphism. $\square$

Hence nontrivial identification of context hierarchies cannot come from categorical equivalence of posets, and the presheaf semantics of Part III does not blur the distinction between different hierarchies. Nontrivial identification must come from functors that coarsen, which are neither full nor faithful, or from relations weaker than equivalence, such as simulation, bisimulation, and observational equivalence. This explains the shape of the results: they are statements of common abstraction together with an analysis of what the abstraction preserves.

The most useful positive result concerns what a state-surjective, observation-reflecting translation does establish.

Theorem 14.6 (Relative equivalence; PROVED HERE). *Let $\Phi = (F, f, \eta, \alpha) : \mathfrak{R} \rightarrow \mathfrak{S}$ be a translation that is state-surjective and observation-reflecting. Then for every context c the map*

$$f_c : Q(c) \rightarrow Q'(Fc), \quad [x]_c \mapsto [f(x)]_{Fc}$$

*is a bijection, and the family (f_c) is a natural isomorphism $Q \cong F^*Q'$ of presheaves on $\mathcal{C}$, where $F^*Q' = Q' \circ F^{\text{op}}$.*

Proof. The map is well defined by Theorem 12.3 and injective because Φ is observation-reflecting. It is surjective because f is. For $u : c \rightarrow d$, $f_c Q(u)[x]_d = [f(x)]_{Fc} = Q'(Fu)f_d[x]_d$. $\square$

Corollary 14.7 (Equivalence relative to retained structure; PROVED HERE). *If $\mathfrak{R}_1$ and $\mathfrak{R}_2$ admit state-surjective, observation-reflecting translations $\Phi_i : \mathfrak{R}_i \rightarrow \mathfrak{S}$ with $F_i = \text{id}$ on a common $\mathcal{C}$, then $Q_{\mathfrak{R}_1} \cong Q_{\mathfrak{R}_2}$ as presheaves on $\mathcal{C}$. The systems are then equivalent as observational presheaves, and may still differ in X , A , and δ , as in Example 13.6.*

This is the form in which a common abstraction can establish equivalence without full equivalence: equivalence of the canonical presheaves, relative to the distinctions that the abstraction retains. The next proposition shows that sharing a common quotient does not even yield mutual translation of the systems.

Proposition 14.8 (Common quotient does not imply mutual translation; PROVED HERE). (a)

There exist systems $\mathfrak{R}_1, \mathfrak{R}_2$ with a common state-surjective, observation-reflecting abstraction $\mathfrak{S}$ such that no translation $\mathfrak{R}_1 \rightarrow \mathfrak{R}_2$ is bijective on states and both preserves and reflects observations.

(b) *There exist such systems with no translation $\mathfrak{R}_1 \rightarrow \mathfrak{R}_2$ at all.*

Proof. (a) Let $\mathcal{C} = 1$ and $\Omega(*) = \{0, 1\}$. Let $X_1 = X_2 = \{1, 2, 3, 4\}$ with o^1 constant 0 on $\{1, 2\}$ and 1 on $\{3, 4\}$, and o^2 constant 0 on $\{1\}$ and 1 on $\{2, 3, 4\}$. Both map by o^i onto $\mathfrak{S}$ with $X' = \{0, 1\}$, $o' = \text{id}$, and are surjective and observation-reflecting. A bijection $h : X_1 \rightarrow X_2$ that both preserves and reflects observations would carry the observation classes $\{1, 2\}, \{3, 4\}$ onto the classes $\{1\}, \{2, 3, 4\}$, which have different sizes. (b) is Example 13.6. $\square$

Without the bijectivity requirement in (a), mutual translations exist: send the class $\{1, 2\}$ to $\{1\}$ and $\{3, 4\}$ into $\{2, 3, 4\}$, and conversely, each preserving and reflecting observations. The requirement is therefore needed to keep constant-like maps from trivializing the example, and part (b) shows that the dynamics supply a genuine obstruction independent of it.

15 Translation Between Mathematical Languages

Set-theoretic, type-theoretic, logical, algebraic, and categorical presentations of mathematical structure are connected by translations of different strengths, and the discussion above supplies a vocabulary for keeping them apart. A *syntactic translation* from a theory T to a theory T' is a compositional map $\varphi \mapsto \varphi^*$ of formulas such that $T \vdash \varphi$ implies $T' \vdash \varphi^*$, and it *reflects provability* when the converse holds. A *semantic interpretation* is a functor between categories of models, typically a reduct. Two theories are *definitionally equivalent* when they have a common definitional extension, and *Morita equivalent* when, after adding new sorts for products, subsorts, and quotients, they are definitionally equivalent [5]. Finally the categories of models may be *equivalent as categories*. These notions are logically independent: a syntactic translation need not reflect provability, a semantic interpretation need not be an equivalence, and categorical equivalence of models need not come from any translation of syntax with preserved proof theory. The following examples show each separation concretely.

Proposition 15.1 (Classical and intuitionistic arithmetic; ESTABLISHED). *Let PA be Peano arithmetic and HA Heyting arithmetic.*

- (a) The Gödel–Gentzen negative translation sends each sentence φ to a sentence φ^N such that $\text{PA} \vdash \varphi$ implies $\text{HA} \vdash \varphi^N$ [38].
- (b) PA is conservative over HA for Π_2^0 sentences [12].
- (c) HA does not prove every theorem of PA: for a Gödel sentence G of PA, $\text{PA} \vdash G \vee \neg G$ while $\text{HA} \not\vdash G \vee \neg G$, assuming PA is consistent.

Proof of (c). Excluded middle gives $\text{PA} \vdash G \vee \neg G$. HA has the disjunction property [38]: if $\text{HA} \vdash A \vee B$ then $\text{HA} \vdash A$ or $\text{HA} \vdash B$. If $\text{HA} \vdash G \vee \neg G$ then, since PA extends HA, $\text{PA} \vdash G$ or $\text{PA} \vdash \neg G$, contradicting the independence of G . $\square$

The same phenomenon appears algebraically. Classical propositional logic corresponds to Boolean algebras and intuitionistic propositional logic to Heyting algebras, and the double-negation map is the algebraic form of the negative translation.

Proposition 15.2 (Glivenko map; ESTABLISHED). *For a Heyting algebra H , the map $a \mapsto \neg\neg a$ is a surjective Heyting algebra homomorphism $H \rightarrow H_{\neg\neg}$ onto the Boolean algebra of regular elements, and it is injective if and only if H is Boolean [13].*

This is a surjective homomorphism of structures that is not an equivalence unless H is already Boolean, an instance of Proposition 14.2: the translation retains the classical theory and discards the intuitionistic distinctions among elements with the same double negation.

There are also genuine equivalences whose presentations differ. Finitary algebraic theories presented as Lawvere theories correspond to finitary monads on **Set**, with equivalent categories of models, although one presentation has operations and equations and the other has none [20]. Typed lambda calculi with products and exponentials, taken up to their equational theories, correspond to cartesian closed categories in an equivalence in which terms are morphisms [18]; this equivalence does not see evaluation strategy or reduction sequences, since it is stated at the level of equality of terms. Rings R and S are Morita equivalent when their module categories are equivalent: a field k and the matrix ring $M_2(k)$ are Morita equivalent and not isomorphic, and the equivalence preserves module-theoretic invariants such as the number of simple modules while losing ring-theoretic invariants such as commutativity. Constructive set theory is interpretable in Martin-Löf type theory [3], and the strength of the type theory underlying contemporary proof assistants has been compared with set theory augmented by inaccessible cardinals [8]; these are results of the form “interpretable in, with a stated strength” and not statements of equivalence. In each case the relation among languages is specified by what the translation preserves.

The correspondence of Part III has a syntactic counterpart in which the orientation duality of Proposition 13.7 reappears. For a model M and a set V of variables, let $X = M^V$ be the set of valuations and let contexts be finite subsets $\Gamma \subseteq V$ with $\Omega(\Gamma) = M^\Gamma$ the restriction of valuations. This is a region system in the sense of Example 9.1. Terms in context Γ form a family $\text{Term}(\Gamma)$ with weakening maps $\text{Term}(\Gamma) \rightarrow \text{Term}(\Delta)$ for $\Gamma \subseteq \Delta$, which run in the direction opposite to restriction, and the evaluation of a weakened term on a valuation equals the evaluation of the original on its restriction. Syntax is covariant and semantics

is contravariant, and the pairing between them is natural. The orientation of information flow is therefore a feature that distinguishes a presentation by terms from a presentation by valuations, as it distinguishes a lexical scope from a filesystem.

Question 15.3 (Interpretations and preservation). To what extent do properties of an interpretation between first-order theories, such as conservativity and bi-interpretability, correspond to properties of the induced translations of valuation systems, such as being observation-reflecting or state-surjective in the sense of Definition 13.1?

16 Universality and Its Limits

Computational universality is a statement that a formalism computes every computable function. The strongest form of the observation that universal systems are interchangeable is Rogers’ isomorphism theorem: any two acceptable numberings of the partial computable functions, which includes the programming systems of any two universal languages, are related by a computable bijection of program indices that preserves the function computed [36]. Universality therefore yields a translation between any two universal systems, and the translation preserves exactly one thing, the input-output function of each program. In the terminology above, it is a surjective representation onto a one-observation system, the weakly terminal abstraction of Proposition 13.3 specialized to the observation “what function is computed”. Nothing else need be preserved, and in general nothing else is.

Several distinct kinds of structure are lost. Resource behavior depends on the model: translating between Turing machines and random-access machines costs a polynomial overhead, so complexity classes defined up to polynomial factors are stable, while fine-grained time bounds are not, and whether the stability extends to every physically realizable model is open, with quantum computation the principal candidate for violating it if the relevant complexity classes differ. Expressive power in a language-design sense is finer than computability: one language is more expressive than another when a construct cannot be eliminated by local rewriting without global reorganization of programs [11]. Definability at higher types separates languages that compute the same first-order functions: the parallel-or operator is not definable in the language PCF, though it is computable, and adding it yields the language for which the standard model is fully abstract [31]. Type systems and proof theories differ for languages with the same computable functions, as the contrast between simply typed and untyped lambda calculi illustrates.

Contextual equivalence provides the bridge to Part III. Let X be a set of program fragments, let the family $\mathcal{U}$ of closing program contexts be a discrete set of observation contexts, and let the observation of a fragment M at a program context $C[\cdot]$ be whether $C[M]$ terminates, valued in $\{0, 1\}$ [25]. Then $\sim_{\mathcal{U}}$ is contextual equivalence. A denotational valuation $\llbracket \cdot \rrbracket : X \rightarrow D$ is sound when $\llbracket M \rrbracket = \llbracket N \rrbracket$ implies $M \sim_{\mathcal{U}} N$, which says that its image-quotient is $\mathcal{U}$ -sound in the sense of Section 11, and it is fully abstract when the converse also holds.

Proposition 16.1 (Full abstraction is the canonical quotient; ESTABLISHED). *The quotient $X \rightarrow X/\sim_{\mathcal{U}}$ by contextual equivalence factors uniquely through every sound surjective valuation*

$X \rightarrow Z$, as $X \rightarrow Z \rightarrow X/\sim_{\mathcal{U}}$. A sound valuation is fully abstract exactly when its kernel is $\sim_{\mathcal{U}}$, that is, when it is itself the terminal $\mathcal{U}$ -sound quotient of Theorem 11.3.

Proof. This is Theorem 11.3(b) applied to the discrete family $\mathcal{U}$, together with the observation that a sound surjective valuation has kernel contained in $\sim_{\mathcal{U}}$, with equality exactly in the fully abstract case. $\square$

The proposition does not claim novelty for the concept of full abstraction, which is classical [31]. It records that full abstraction is an instance of the canonical quotient of the preceding sections, and that the universal property of Theorem 11.3 is a statement about every sound semantics at once.

Proposition 16.2 (Universality does not fix the observation; ESTABLISHED). *Let PCF and PCF + por denote the typed functional language with the usual arithmetic and fixed-point constructs and the same language extended by parallel-or. Both languages are universal for the first-order computable functions on the natural numbers, yet their contextual equivalences on higher-type terms differ, so the two canonical quotients of Theorem 11.3 are distinct.*

Proof. The parallel-or operator is not definable in PCF [31], and terms of higher type that no PCF context can separate are separated by a context that uses it. The family of closing contexts of the extended language therefore properly contains that of the base language on the shared terms, and the corresponding observational kernel is strictly finer. $\square$

Finally, universality interacts with contexts in a specific way. Closure conversion and defunctionalization translate programs in a lexically scoped higher-order language into a first-order language with a single level of scope, preserving the function computed [35, 23]. The nested scope tree of the source disappears as a context structure and reappears as data, in the form of explicit environment records passed as arguments.

Proposition 16.3 (Contexts migrate into data; ESTABLISHED). *Closure conversion and defunctionalization are translations from lexically scoped higher-order programs to first-order programs that preserve the function computed [35, 23]. In the output of either translation the source’s scope tree is not represented as a context structure of the target, and the nesting survives only as the structure of data, namely explicit environment records.*

The consequence for structural comparison is that the two cited translation methods do not preserve contexts as contexts, although each preserves an encoding of them; whether some other function-preserving translation could retain the context structure explicitly in its target is not excluded by the argument. Whether a common structural representation can exist even when strong equivalence fails then has a definite answer. Theorem 13.4 supplies one for any fixed observation presheaf Ω , namely the set of global sections, and it applies to universal languages as to any others. What universality does not supply is a choice of Ω , and the structure retained is exactly the structure that Ω carries.

Part V

Results and Interpretation

17 A General Contextual Translation Construction

The preceding sections have introduced the components one at a time. Their assembly into a single construction is the following. A contextual representation system $\mathfrak{R} = (\mathcal{C}, X, \Omega, \mathcal{A}, \delta)$ supplies contexts and refinement arrows, local information assignments in the presheaf Ω , observation and disclosure through the maps o_c and their refinements, admissible operations through $\mathcal{A}$ and δ , and structure-preserving translations through **CRS**. Each component carries assumptions, and the results depend on different subsets of them. Smallness of $\mathcal{C}$ is needed for Ω to be a presheaf in **Set** and for $\Gamma(\Omega)$ to be a set. Coherence $o_c = \Omega(u)o_d$ is needed for every result about kernels, for the canonical presheaf Q , and for the profile map. Surjectivity of the observation maps is a normalization that can always be achieved by replacing Ω with Q , and it is what makes the observational structure equivalent to a family of partitions. Determinism and partiality of δ are needed for behavioral minimization and for the simulation condition (D). No assumption at all is placed on $\mathcal{A}$, which is why authority is independent of observation. The table records which results use which assumptions.

Result	coherence	surjective o_c	δ	$\mathcal{A}$
Observational refinement (Thm. 3.3)	yes	no	no	no
Canonical presheaf (Thm. 11.1)	yes	no	no	no
Coarsest quotient (Thm. 11.3)	yes	no	no	no
Terminal observed system (Thm. 13.4)	yes	no	no	no
Partition representation (Thm. 17.1)	yes	yes	no	no
Tree representation (Thm. 17.2)	yes	yes	no	no
Behavioral minimization (Prop. 11.5)	yes	no	yes	no
Safety transfer (Prop. 18.2)	yes	no	yes	yes

The first representation theorem shows that the whole observational content of a system, up to equivalence of categories, is a family of equivalence relations indexed by the contexts. Let **Obs** be the category whose objects are quadruples $(\mathcal{C}, X, \Omega, o)$ with coherent observations and every o_c surjective, and whose morphisms are triples (F, f, η) satisfying (T1) to (T3) and (O). Let **Part** be the category whose objects are triples $(\mathcal{C}, X, \approx)$ in which $\approx$ assigns to each context c an equivalence relation $\approx_c$ on X with $\approx_d \subseteq \approx_c$ for every arrow $c \rightarrow d$, and whose morphisms are pairs (F, f) with $F : \mathcal{C} \rightarrow \mathcal{C}'$ a functor and $f : X \rightarrow X'$ a map such that $x \approx_c y$ implies $f(x) \approx'_{F(c)} f(y)$.

Theorem 17.1 (Partition representation; PROVED HERE). *The functor $K : \mathbf{Obs} \rightarrow \mathbf{Part}$ sending $(\mathcal{C}, X, \Omega, o)$ to $(\mathcal{C}, X, \sim)$ and (F, f, η) to (F, f) is an equivalence of categories.*

Proof. K is well defined on objects by Theorem 3.3 and on morphisms by Theorem 12.3. It is faithful because η_c is determined by f : surjectivity of o_c and (O) force $\eta_c(o_c(x)) = o'_{F(c)}(f(x))$.

It is full: given (F, f) in **Part**, define $\eta_c(o_c(x)) = o'_{Fc}(f(x))$, which is well defined because $o_c(x) = o_c(y)$ means $x \sim_c y$, hence $f(x) \sim'_{Fc} f(y)$, which is $o'_{Fc}f(x) = o'_{Fc}f(y)$. Naturality of η follows from coherence of o and o' and surjectivity of o_d , since $\eta_c\Omega(u)o_d(x) = \eta_c o_c(x) = o'_{Fc}f(x) = \Omega'(Fu)o'_{Fd}f(x) = \Omega'(Fu)\eta_d o_d(x)$. It is essentially surjective: given $(\mathcal{C}, X, \approx)$, the system with $\Omega(c) = X/\approx_c$, restriction maps the canonical projections, and o_c the quotient maps, is coherent and is sent by K to $(\mathcal{C}, X, \approx)$. $\square$

The theorem identifies the common abstraction of all coherent observation structures: a set with a contravariant monotone assignment of partitions to contexts, that is, a functor $\mathcal{C}^{\text{op}} \rightarrow \text{Eq}(X)$ into the lattice of equivalence relations on X ordered by inclusion. This is the canonical presheaf of Theorem 11.1 seen from the side of the state set.

A second representation theorem treats the two tree-shaped classes of Section 13, and answers the question of when a presheaf on a tree is a region system or an accumulation system for some decoration. Let T be a finite rooted tree with $\mathcal{C}^\downarrow = (T, \preceq)$ and $\mathcal{C}^\uparrow = (T, \preceq)^{\text{op}}$ as before, and let (T, L) be a decorated tree with all $L(p)$ nonempty. For a presheaf Ω on $\mathcal{C}^\downarrow$, the restriction maps are $\Omega(p) \rightarrow \Omega(q)$ for $q \preceq p$, and for a node p with children $c_1, \dots, c_k$ let $\rho_p : \Omega(p) \rightarrow \prod_i \Omega(c_i)$ be the joint restriction. For a presheaf Ω on $\mathcal{C}^\uparrow$, the restriction maps are $\Omega(q) \rightarrow \Omega(p)$ for $q \preceq p$, and for a non-root node q with parent p let $r_q : \Omega(q) \rightarrow \Omega(p)$ be the restriction to the parent. A map of sets has *uniform fibers* if all its fibers have the same cardinality.

The theorem concerns specified observational models built over a common decorated tree, namely the region and accumulation systems of Section 13. It is not a translation between filesystem semantics and lexical-scope semantics, which differ further in path resolution, shadowing, execution, permissions, and state change, and it should not be read as an equivalence of those disciplines.

Theorem 17.2 (Representation of tree systems; PROVED HERE). *(a) A presheaf Ω on $\mathcal{C}^\downarrow$ with $\Omega(p) \neq \emptyset$ at each leaf p is isomorphic to $\Omega^\downarrow(T, L)$ for some decoration L if and only if, for every node p with children, the map ρ_p is surjective with uniform fibers. The cardinality of $L(p)$ is then the common fiber size of ρ_p , and $|L(p)| = |\Omega(p)|$ for a leaf.*

(b) A presheaf Ω on $\mathcal{C}^\uparrow$ with $\Omega(\text{root}) \neq \emptyset$ is isomorphic to $\Omega^\uparrow(T, L)$ for some decoration L if and only if, for every non-root node q , the map r_q is surjective with uniform fibers. The cardinality of $L(q)$ is then the common fiber size of r_q , and $|L(\text{root})| = |\Omega(\text{root})|$.

Proof. (a) If $\Omega = \Omega^\downarrow(T, L)$ then $\Omega(p) = L(p) \times \prod_i \Omega(c_i)$ because the nodes nested in p are p together with the nodes nested in each child, and ρ_p is the projection, which is surjective with every fiber in bijection with $L(p)$. Conversely, assume the fiber condition. Proceed by induction on the height of the subtree. For a leaf p take $L(p) = \Omega(p)$ and $\theta_p = \text{id}$. For a node p with children choose a set $L(p)$ whose cardinality is the fiber size of ρ_p , and for each $y \in \prod_i \Omega(c_i)$ choose a bijection $\beta_y : \rho_p^{-1}(y) \rightarrow L(p)$. Define $\theta_p : \Omega(p) \rightarrow \prod_{q \preceq p} L(q)$ by

$$\theta_p(z) = (\beta_{\rho_p(z)}(z), \theta_{c_1}(z|_{c_1}), \dots, \theta_{c_k}(z|_{c_k})),$$

where $z|_{c_i}$ is the restriction of z to c_i . Given a value $(\ell, w_1, \dots, w_k)$, the element $y = (\theta_{c_i}^{-1}(w_i))_i$ is determined by the inductive hypothesis, and $z = \beta_y^{-1}(\ell)$ is the unique preimage, so θ_p is bijective. Naturality for $q \preceq p$ follows by induction, because the restriction from p to q factors through a child of p and θ_p contains the components θ_{c_i} of the children. (b) If $\Omega = \Omega^\uparrow(T, L)$ then $\Omega(q) = L(q) \times \Omega(p)$ for the parent p , and r_q is the projection, which is surjective with fibers in bijection with $L(q)$. Conversely, take $L(\text{root}) = \Omega(\text{root})$, and proceed downward from the root. For a node q with parent p choose $L(q)$ of the fiber cardinality of r_q , and for each $y \in \Omega(p)$ choose a bijection $\beta_y : r_q^{-1}(y) \rightarrow L(q)$. Define $\theta_q(z) = (\beta_{r_q(z)}(z), \theta_p(r_q(z))) \in L(q) \times \prod_{p' \preceq p} L(p')$, which is bijective by the same argument and natural because restriction from q to any ancestor is a composite of restrictions to parents. $\square$

Corollary 17.3 (Common abstraction of filesystem and lexical scope; PROVED HERE). *A region system and an accumulation system over the same tree T whose decorations have the same cardinalities are both recovered from the single datum $(T, |L|)$, the rooted tree with a cardinal at each node, by two different functors: the product over nodes nested in a context and the product over ancestors of a context. The decorated tree is canonical, since the cardinalities are recovered as fiber sizes by Theorem 17.2. The orientation of information flow is not part of the common datum, and by Proposition 13.7 it cannot be recovered by an isomorphism of context categories when T branches.*

This is the representation proposition for a specified class: tree-shaped coherent systems with uniform local structure are decorated trees, in two dual forms. It makes the thesis of Section 1 precise for the filesystem and the lexical scope, whose common abstraction is the decorated tree and whose differences are the orientation of the product and the dynamics $\mathcal{A}$ and δ that the abstraction does not carry.

Canonical abstraction is also functorial in the system, which is what allows it to be used for comparison.

Proposition 17.4 (Functoriality of the canonical quotient; PROVED HERE). *Let $\Phi = (F, f, \eta, \alpha) : \mathfrak{R} \rightarrow \mathfrak{S}$ be a translation, let $\mathcal{U}$ be a family of contexts of $\mathfrak{R}$, and let $\mathcal{U}'$ be a family of contexts of $\mathfrak{S}$ with $\mathcal{U}' \subseteq F(\mathcal{U})$. Then there is a unique map $f_{\mathcal{U}} : X_{\mathcal{U}} \rightarrow X'_{\mathcal{U}'}$ with $q'_{\mathcal{U}'} f = f_{\mathcal{U}} q_{\mathcal{U}}$. If Φ is state-surjective and observation-reflecting and $F(\mathcal{U}) = \mathcal{U}'$, then $f_{\mathcal{U}}$ is a bijection.*

Proof. If $x \sim_{\mathcal{U}} y$ then $x \sim_c y$ for all $c \in \mathcal{U}$, so $f(x) \sim'_{F c} f(y)$ by Theorem 12.3, hence $f(x) \sim'_{c'} f(y)$ for all $c' \in F(\mathcal{U}) \supseteq \mathcal{U}'$, and so $f(x) \sim'_{\mathcal{U}'} f(y)$. This gives well-definedness, and uniqueness follows from surjectivity of $q_{\mathcal{U}}$. For the second claim, $f_{\mathcal{U}}$ is surjective because f is, and if $f(x) \sim'_{\mathcal{U}'} f(y)$ with $\mathcal{U}' = F(\mathcal{U})$, then $f(x) \sim'_{F c} f(y)$ for every $c \in \mathcal{U}$, hence $x \sim_c y$ for every $c \in \mathcal{U}$ by reflection. $\square$

Two problems remain open at the level of the construction. A correspondence between two systems, specified as a span $\mathfrak{R}_1 \leftarrow \mathfrak{M} \rightarrow \mathfrak{R}_2$, determines which parts are to be identified, and the finest common abstraction identifying them would be the pushout. Whether pushouts exist for spans of observation-reflecting translations is not established here.

Question 17.5 (Pushouts of correspondences). Does **Part**, and hence **Obs**, have pushouts of spans in which one leg is observation-reflecting, and when they exist, do they have a description in terms of generated equivalence relations on the amalgamated state set?

18 Preservation and Loss Under Translation

This section assembles the answers to the question of what survives translation, separating results proved from conjectures and illustrative correspondences. Each of the properties named in the outline admits a precise statement.

Contextual reachability. A translation preserves reachability and need not reflect it. Say a translation is *bounded* if for every state x and every action $a' \in \Sigma'$ with $\delta'(f(x), a')$ defined, there is $a \in \Sigma$ with $\alpha(a) = a'$ and $\delta(x, a)$ defined and $f(\delta(x, a)) = \delta'(f(x), a')$.

Proposition 18.1 (Reachability; PROVED HERE). *Let $\Phi : \mathfrak{R} \rightarrow \mathfrak{S}$ be a translation. If y is reachable from x in $\mathfrak{R}$ then $f(y)$ is reachable from $f(x)$ in $\mathfrak{S}$. If Φ is bounded, then conversely, every state z' reachable from $f(x)$ in $\mathfrak{S}$ is of the form $f(y)$ for some y reachable from x in $\mathfrak{R}$. Boundedness cannot be dropped.*

Proof. The first claim follows from (D) by induction on the length of a run, and the second by the same induction using boundedness. For the failure, let $\mathfrak{R}$ have states $\{0, 1\}$ and no transitions, let $\mathfrak{S}$ have states $\{0, 1\}$ and one transition $\delta'(0, a) = 1$, and let $f = \text{id}$ with the remaining components trivial. Condition (D) holds vacuously, the state 1 is reachable from $f(0) = 0$ in $\mathfrak{S}$, and 1 is not reachable from 0 in $\mathfrak{R}$, so boundedness fails and reachability is not reflected. $\square$

Observable distinctions. Theorem 12.3 states that distinctions of the source are not created and Proposition 12.4 characterizes when they are reflected: exactly when η_c is injective on the image of o_c . Theorem 14.6 states that for state-surjective, observation-reflecting translations the canonical presheaves are isomorphic, and Corollary 13.5 identifies what the canonical abstraction forgets.

Observationally equivalent contexts. Two contexts c, c' of a system are *observationally equivalent* if $\sim_c = \sim_{c'}$, equivalently if $Q(c)$ and $Q(c')$ are canonically isomorphic as quotients of X . Contexts that are isomorphic in $\mathcal{C}$ are observationally equivalent, by Theorem 3.3 applied in both directions, and the converse fails: two distinct directories that are aliases of one another through a bind mount, or two rooms that a game always keeps in the same state, are observationally equivalent without being connected by arrows. The criterion that identifies them is equality of kernels, which is the partition representation of Theorem 17.1, and passing to the quotient identifies them without changing the observational structure.

Authority and prohibition. Condition (A) is lax, and its consequences are one-sided. To state them it is necessary to say how a state determines its current context. Call a system *positioned* if it carries a map $\text{pos} : X \rightarrow \text{Ob}(\mathcal{C})$, and say that a transition $x \xrightarrow{a} \delta(x, a)$ is *admissible* if $a \in \mathcal{A}(\text{pos}(x))$. A translation of positioned systems is positioned if $F \circ \text{pos} = \text{pos}' \circ f$.

Proposition 18.2 (Authority, prohibition, and safety; PROVED HERE). *Let $\Phi : \mathfrak{R} \rightarrow \mathfrak{S}$ be a positioned translation.*

- (a) *Admissible transitions of $\mathfrak{R}$ are sent to admissible transitions of $\mathfrak{S}$: no permitted behavior is lost.*
- (b) *(Safety transfer.) Let $B' \subseteq X'$ be a set of bad states and $B = f^{-1}(B')$. If no admissible run of $\mathfrak{S}$ from $f(x_0)$ reaches B' , then no admissible run of $\mathfrak{R}$ from x_0 reaches B .*
- (c) *Prohibitions are not preserved. There exist a positioned translation $\mathfrak{R} \rightarrow \mathfrak{S}$ and a state set B' such that $B = f^{-1}(B')$ is unreachable in $\mathfrak{R}$ and B' is reachable in $\mathfrak{S}$.*

Proof. (a) If $x \xrightarrow{a} y$ is admissible then $\delta'(f(x), \alpha(a)) = f(y)$ by (D) and $\alpha(a) \in \alpha(\mathcal{A}(\text{pos}(x))) \subseteq \mathcal{A}'(F\text{pos}(x)) = \mathcal{A}'(\text{pos}'(f(x)))$ by (A) and positionedness. (b) is the contrapositive of the lifting of an admissible run of $\mathfrak{R}$ to an admissible run of $\mathfrak{S}$ by part (a). (c) Let $\mathfrak{R}$ have a single context c with $\mathcal{A}(c) = \{\text{use}\}$, states $\{0, 1\}$ and actions $\{\text{use}, \text{admin}\}$, with $\delta(0, \text{use}) = 0$ and $\delta(0, \text{admin}) = 1$, and initial state 0. The action **admin** is not admissible, so the state 1 is unreachable by admissible runs. Let $\mathfrak{S}$ be the same system with $\mathcal{A}'(c) = \{\text{use}, \text{admin}\}$ and Φ the identity on everything else, which satisfies (A) and (D). The state 1 is reachable in $\mathfrak{S}$. $\square$

Hence a translation that forgets authority over-approximates behavior and loses precisely the safety properties that rest on prohibitions: the coarse image can be used to prove safety of the source only if it already proves it, and it fails to prove it when the safety of the source depends on what the source forbids. For the general access-matrix model, the question whether a right can leak is undecidable [17], so no uniform procedure computes the loss.

Global reconstruction. For translations that are the identity on a common site, the relation between the gluing behavior of source and target is as follows.

Proposition 18.3 (Reconstruction under translation; PROVED HERE). *Let $\Phi : \mathfrak{R} \rightarrow \mathfrak{S}$ be a translation with $F = \text{id}$ on a common context category and a covering sieve S .*

- (i) *If Φ is state-surjective and observation-reflecting then $Q \cong Q'$, so Q is a sheaf for S exactly when Q' is.*
- (ii) *If Φ is observation-reflecting, the induced map $Q \rightarrow Q'$ is a monomorphism, so if Q' is separated for S then Q is separated for S .*
- (iii) *Observation-reflecting, state-injective translations need not reflect gluing: there are $\mathfrak{R}, \mathfrak{S}$ with Q' a sheaf for S and Q not.*

(iv) *State-surjective translations that are not observation-reflecting need not reflect separation: there are $\mathfrak{R}, \mathfrak{S}$ with Q' a sheaf for S and Q not separated.*

Proof. (i) is Theorem 14.6. (ii) The maps $f_c : Q(c) \rightarrow Q'(c)$ are injective by reflection and natural, and a subpresheaf of a separated presheaf is separated, since two elements with equal restrictions to the cover have equal images in the target and hence are equal. (iii) Take $\mathfrak{R}$ to be the system of acyclic orientations of Example 10.4, $\mathfrak{S}$ the system of all eight orientations with the same Ω , and f the inclusion. The map is injective and observation-reflecting, every matching family for the cover by singletons is realized in $\mathfrak{S}$, and gluing fails in $\mathfrak{R}$. (iv) Take $\mathfrak{R}$ to be Example 10.5 and $\mathfrak{S}$ the system obtained by discarding the coordinate p and the corresponding component of $\Omega(L)$, with f the projection. The map is surjective and not observation-reflecting at L , and Q' is a sheaf for the cover by $\{a\}$ and $\{b\}$ while Q is not separated. $\square$

Status. The preservation results above are proved. Conjectures concerning the range of languages covered by the construction are collected in Section 19 and in the inventory of Appendix D. The computational cost of deciding whether a translation exists between two finite systems is not addressed by any result here.

Question 18.4 (Complexity of translation). What is the computational complexity of deciding, for finite systems $\mathfrak{R}$ and $\mathfrak{S}$, whether a translation $\mathfrak{R} \rightarrow \mathfrak{S}$ exists, under each of the variants of Definition 12.1?

19 The Limits of Hierarchical Representation

A tree is the most economical structure for nested contexts, supporting induction on structure, a unique chain of ancestors, and canonical forms. Several features of the systems examined here are not trees, and each one represents information that a hierarchy either discards or must reintroduce by additional structure.

Overlapping scopes and multiple parents. A context with two incomparable containers, as in the diamond of Proposition 6.5, has no unique chain of ancestors, so resolution needs a policy that linearizes the containers. Python’s attribute lookup for classes uses the C3 linearization, so that for classes $A, B(A), C(A)$, and $D(B, C)$ the lookup order for D is D, B, C, A , `object` [27]. The linearization is an additional datum, not a consequence of the poset, and it can fail to exist: declaring a class with bases A and then $B(A)$ in that order is rejected because no linear order is consistent with the constraints. A hierarchy of classes underdetermines resolution, and the underdetermination is resolved by structure the hierarchy does not contain.

Cycles. Mutually dependent modules form a cycle in the import graph. The reflexive transitive closure collapses a cycle to an equivalence class, which retains reachability and loses the order of entry, yet the behavior of a cyclic import depends on which module is entered

first, since the second sees a partially initialized first. The behavior depends on the route and not only on the reachability relation, which is the distinction of Proposition 2.3.

Non-tree navigation and state-dependent accessibility. Portals and doors connect contexts in ways that do not respect any nesting, and the edge from one context to another may exist only in some states, so that the ungarded context graph over-approximates by Proposition 8.1.

Dynamically generated contexts. When contexts are created by evaluation of code at run time or by procedural generation of levels, the set of contexts is a function of the state, and a representation with a fixed small category $\mathcal{C}$ is not available. The remedy is to take $\mathcal{C}$ to be the category of all potential contexts, with an observation that records whether a context currently exists, which moves the dynamics into the observation.

Nonlocal constraints. A constraint relating contexts in different branches, such as global uniqueness of names or acyclicity of dependencies, is not expressible along the edges of a tree, and Example 10.4 exhibits its effect as a failure of gluing.

The case of non-tree navigation can be treated exactly. For a rooted graph, the *unfolding* is the tree whose nodes are finite directed paths from the root, with the parent of a path being the path with its last edge removed, and the map π sends a path to its last vertex.

Proposition 19.1 (Unfolding is a bisimulation; ESTABLISHED). *For a rooted labeled transition system $(S, s_0, \rightarrow)$, the unfolding T with edges labeled by the last transition is a tree-shaped system, and the relation $\{(p, \pi(p))\}$ is a bisimulation between T and S . In particular the unfolding preserves traces and branching behavior [37].*

Proposition 19.2 (Information lost by unfolding; PROVED HERE). *Let $\pi : T \rightarrow \Gamma$ be the unfolding of a rooted graph.*

- (a) *π is injective on nodes if and only if the part of Γ reachable from the root is a tree.*
- (b) *If a cycle is reachable, then T is infinite.*
- (c) *If a vertex is reachable by two distinct routes then its preimage has at least two elements, so the sharing of that vertex is not visible in the structure of T .*

Proof. Distinct paths with the same endpoint give distinct nodes with the same image, which is (a) and (c). A reachable cycle can be traversed arbitrarily often, giving infinitely many paths, which is (b). $\square$

The preservation of bisimilarity in Proposition 19.1 applies to the transition system whose configurations include all state, and it does not extend to a representation in which state is attached to the nodes of the tree. Placing a local state at each node of the unfolding creates independent copies of the state of a shared context. The condition that restores the system is descent.

Proposition 19.3 (Descent; PROVED HERE). *Let $R(\gamma)$ be a set of local values at each vertex γ of Γ . A state on Γ is a family $x \in \prod_{\gamma} R(\gamma)$, and its pullback along π assigns $x(\pi(p))$ to each node p of T . A family $y \in \prod_{p \in T} R(\pi(p))$ is a pullback if and only if it is constant on the fibers of π .*

The constraint of constancy on fibers relates nodes in different branches of the tree and is not expressible by conditions along the edges of the tree. The following example shows that omitting it changes behavior.

Example 19.4 (Shared room reached by two routes). Let Γ have vertices h, a, b, r and edges $h \rightarrow a, h \rightarrow b, a \rightarrow r, b \rightarrow r$ together with the reverse edges, and let the room r carry a bit $z \in \{0, 1\}$ initially 0, with an action `toggle` available at r that flips z and an observation at r that reveals z . In the system over Γ with the bit in the configuration, the run that goes $h \rightarrow a \rightarrow r$, toggles, returns to h , and goes $h \rightarrow b \rightarrow r$ observes $z = 1$. In the tree T with a local bit at each node, the nodes har and hbr carry independent bits, so the same sequence of actions observes $z = 0$ at hbr . The two systems are therefore not bisimilar, although the unfolding of the configuration graph of Γ , with the bit in the configuration, is bisimilar to it by Proposition 19.1. The discrepancy is exactly the failure of descent.

Conjecture 19.5 (Descent for dynamics). Let $\pi : T \rightarrow \Gamma$ be the unfolding of a finite rooted graph. Let $\mathbf{Desc}(\pi)$ be the category whose objects are contextual representation systems over T satisfying descent along π at the level of states, in the sense of Proposition 19.3, and whose transition function factors through the quotient of the state set by the fiber identifications, with the corresponding compatibility of the observation maps and of the authority assignment $\mathcal{A}$ with those identifications; morphisms are translations that are identities on contexts. Then $\mathbf{Desc}(\pi)$ is equivalent to the category of contextual representation systems over Γ with the same morphisms.

The conjecture is the extension of Proposition 19.3 from states to dynamics. The state-level statement is proved, and the dynamics require a precise definition of equivariance that has not been supplied here. In summary, a tree representation is useful without being structurally complete: it preserves behavior up to bisimulation when state lives in the configuration, and it loses identity of repeated contexts, finiteness in the presence of cycles, and the nonlocal constraints that must be reintroduced as descent, guards, stacks, or linearizations.

20 Conclusion: Equivalence Relative to Abstraction

The central question was under what conditions heterogeneous formal systems can be represented by a shared structure of contexts, observations, and admissible transformations. The development has given the question a determinate answer in four parts. First, a common abstraction always exists in a trivial sense and is informative only relative to what it retains, since there is a terminal system that retains nothing and a disjoint union that retains everything while identifying nothing (Proposition 13.3). Second, relative to a chosen family of observations, a canonical abstraction exists and is characterized by universal properties:

the canonical presheaf of observational distinctions is the coarsest presheaf quotient through which observations factor (Theorem 11.1), the quotient of states by a family of contexts is the coarsest compatible with the family (Theorem 11.3), and the set of global sections is the terminal system observed through a fixed presheaf (Theorem 13.4). Third, translations preserve observational distinctions always and reflect them exactly when the observation map is injective on its image (Theorem 12.3 and Proposition 12.4), preserve permitted behavior while discarding prohibitions (Proposition 18.2), and preserve reachability when bounded (Proposition 18.1). Fourth, surjective and even faithful representations establish equivalence only of the canonical presheaves (Theorem 14.6), and the examples show that this leaves the dynamics, the authority structure, the multiplicity of microstates, and the orientation of information flow free to differ (Example 13.6, Proposition 13.7).

The difference between the existence of a common abstraction and the claim that all mathematical languages are equivalent is therefore a difference of quantifiers over structure. A common abstraction asserts that some structure is shared. Equivalence asserts that all relevant structure is shared, and a theory of what is relevant is what a notion of equivalence supplies: categorical equivalence of posets collapses to isomorphism (Proposition 14.4), equivalence of presheaf categories does the same (Corollary 14.5), translation between logics may preserve provability in one direction without reflecting it (Proposition 15.1), and computational universality equates languages only at the level of the function computed (Section 16). The statement that survives is the one with which the essay began: distinct mathematical languages can possess a canonical common abstraction relative to a specified family of observable structures, while their differences are characterized by the information and operations that the abstraction necessarily forgets. The statement is not a theorem about all languages; it is a method, and its range is the class of systems that can be presented as contextual representation systems, which includes the filesystems, lexical scope disciplines, access-control models, and transition systems treated here, and whose boundary is not yet known.

Several directions follow. In categorical semantics, the category **CRS** and its observational reduct **Part** invite a study of limits and colimits, beginning with Question 17.5, and of an internal logic for contextual systems in which the sheaf condition and its failures are logical notions. In contextual logic, the independence of knowledge, visibility, and authority suggests a modal treatment in which the three assignments are separate modalities, and the commutation criterion of Proposition 4.1 becomes an equation between modalities. In programming-language theory, the observation that full abstraction is the canonical quotient (Proposition 16.1) suggests comparing scope disciplines by the kernel families they induce, and the representation theorem for trees (Theorem 17.2) suggests extending decorated trees to scope graphs with the guards and linearizations of Section 19. In mathematical representation, the relation of the present construction to frameworks for translation among logics, such as institutions, is outlined in Appendix E and remains to be worked out, as do Question 15.3 and the relation to simulation-based accounts of reasoning in Question E.1. Throughout, the results are stated with their hypotheses, and the conjectures are marked as such. The strongest version of the claim that heterogeneous formal systems are one structure is not supported by the mathematics, and the weaker versions are supported to exactly the degree

that the retained structure is specified.

A Definitions and Notation

A *category* $\mathcal{C}$ consists of objects, arrows $u : c \rightarrow d$ with composition vu and identities id_c . A *functor* $F : \mathcal{C} \rightarrow \mathcal{D}$ preserves identities and composition. A *natural transformation* $\eta : F \Rightarrow G$ between functors $F, G : \mathcal{C} \rightarrow \mathcal{D}$ is a family $\eta_c : Fc \rightarrow Gc$ with $\eta_d F(u) = G(u) \eta_c$ for $u : c \rightarrow d$. A *presheaf* on $\mathcal{C}$ is a functor $\mathcal{C}^{\text{op}} \rightarrow \mathbf{Set}$, and the presheaf category $[\mathcal{C}^{\text{op}}, \mathbf{Set}]$ has natural transformations as morphisms. The constant presheaf on a set X is written ΔX . Epimorphisms and monomorphisms of presheaves are the componentwise surjective and injective natural transformations, and every natural transformation factors as an epimorphism followed by a monomorphism, the factorization being unique up to unique isomorphism.

A functor is *full* if surjective on hom-sets, *faithful* if injective on hom-sets, and *essentially surjective* if every object is isomorphic to one in the image. An *equivalence of categories* is a functor with a quasi-inverse, and Theorem 14.1 characterizes it. A preorder is *skeletal* if antisymmetric. The *Cauchy completion* of a small category is obtained by splitting idempotents.

For a presheaf P on $\mathcal{C}$, a *sieve* on c is a set of arrows into c closed under precomposition, a *Grothendieck topology* selects covering sieves subject to the axioms of maximality, stability under pullback, and transitivity, and the sheaf condition for a covering sieve is that every matching family has a unique amalgamation. The *limit* $\lim P$ of a presheaf viewed as a diagram is the set of compatible families $(s_c)_c$ with $P(u)(s_d) = s_c$.

An *equivalence relation* on X is a reflexive, symmetric, transitive relation, and the set $\text{Eq}(X)$ of such relations is a complete lattice under inclusion. The *kernel* of a map g is $\{(x, y) : g(x) = g(y)\}$. A *labeled transition system* is $(S, \Sigma, \rightarrow)$ with $\rightarrow \subseteq S \times \Sigma \times S$, a *simulation* from S to S' is a map or relation sending each transition to a transition, and a *bisimulation* is a relation R such that both R and R^{-1} are simulations. Partial maps are written $\rightarrow$, and a partial map g is *defined at* x if $x \in \text{dom } g$.

B Finite Examples and Counterexamples for Visibility, Disclosure, and Access Control

B.1 Commutation of navigation and disclosure

Let $t = (c, \text{go}, c')$ and $S = \{0, 1\}$ (or $\{0, 1\}^2$ below). Entries give the configuration reached, and a dash-free “undefined” marks the failure of a composite to be defined.

Setting	(c, s)	$N_t D_f(c, s)$	$D_f N_t(c, s)$
global toggle, $\rho_t = \text{id}$	$(c, 0)$	$(c', 1)$	$(c', 1)$
per-folder toggle, $S = \{0, 1\}^2$, $\rho_t = \text{id}$	$(c, (0, 0))$	$(c', (1, 0))$	$(c', (0, 1))$
reset on entry, $\rho_t \equiv 0$, toggle	(c, s)	$(c', 0)$	$(c', 1)$
virtual target, $f_{c'}$ undefined	$(c, 0)$	$(c', 1)$	undefined

The four rows are the cases of Proposition 4.1: the first satisfies $\rho_t f_c = f_{c'} \rho_t$ and the other three violate it.

B.2 Authority and the three orders

Context	Working directory	Spawned by	$\mathcal{A}$	Delegation
u	/home	(login)	{read, write}	top
l	/home/sandbox	u	{read}	$l \sqsubseteq u$
s	/home/bin	u	{read, write, admin}	none

Containment gives $l \preceq u$ and $s \preceq u$, spawn gives l and s as children of u , and $\mathcal{A}$ is monotone for delegation alone.

B.3 State-dependent accessibility

The system $S = \{A, B\} \times \{0, 1\}$ with initial state $(A, 0)$, transitions $(A, 0) \xrightarrow{\text{take}} (A, 1)$ and $(A, 1) \xrightarrow{\text{go}} (B, 1)$, and $q(r, k) = r$ has context graph with edges $A \xrightarrow{\text{take}} A$ and $A \xrightarrow{\text{go}} B$. From the initial context A , the quotient admits the run go , which the system does not from $(A, 0)$.

B.4 Shadowing and coherence

Take states $(x_p, x_q) \in \{0, 1\}^2$ for an outer scope p and an inner scope q that rebinds the same name. The flat observations $o_p = x_p$ and $o_q = x_q$ have kernels with classes $\{(0, 0), (0, 1)\}, \{(1, 0), (1, 1)\}$ at p and $\{(0, 0), (1, 0)\}, \{(0, 1), (1, 1)\}$ at q , which are incomparable, so coherence fails for the arrow $p \rightarrow q$. The frame observations $o_p = x_p$ and $o_q = (x_p, x_q)$ have kernels with two classes at p and four at q , nested as required.

C Worked Translations Among Filesystem, Lexical-Scope, and Transition-System Models

Let T be the rooted tree with root R , children A and B , and a node C whose parent is A , so that $C \preceq A \preceq R$ and $B \preceq R$. Let every local set be $L(p) = \{0, 1\}$, so that $X = \{0, 1\}^4$ has 16 states with coordinates (x_R, x_A, x_B, x_C) .

Filesystem as region system. The context category is $\mathcal{C}^\downarrow$ with arrows $C \rightarrow A \rightarrow R$ and $B \rightarrow R$. The observation spaces are the products over nested nodes, and the kernels are as follows.

Context p	$\Omega^\downarrow(p)$	Classes of $\sim_p^\downarrow$	Agree on
R	$\{0, 1\}^4$	16	R, A, B, C
A	$\{0, 1\}^2$	4	A, C
B	$\{0, 1\}$	2	B
C	$\{0, 1\}$	2	C

The family $\{R\}$ is generating, and $X_{\{R\}} = X$ has 16 classes.

Lexical scope as accumulation system. The context category is $\mathcal{C}^\uparrow$ with arrows $R \rightarrow A \rightarrow C$ and $R \rightarrow B$, and the observation spaces are the products over ancestors.

Context q	$\Omega^\uparrow(q)$	Classes of $\sim_q^\uparrow$	Agree on
R	$\{0, 1\}$	2	R
A	$\{0, 1\}^2$	4	R, A
B	$\{0, 1\}^2$	4	R, B
C	$\{0, 1\}^3$	8	R, A, C

The family $\{C\}$ alone is not generating, because B is not a restriction of C , and $X_{\{C\}}$ has 8 classes, identifying states that differ only in x_B . The family $\{B, C\}$ of innermost scopes is generating, and $X_{\{B, C\}} = X$ has 16 classes, as Theorem 11.3(c) predicts. The two kernel families are not related by any isomorphism of the indexing categories, since $\mathcal{C}^\downarrow$ has one maximal element and two minimal ones, and $\mathcal{C}^\uparrow$ has two maximal and one minimal.

Game world as transition system. Let the configurations be $S = \{R, A, B, C\} \times \{0, 1\}^4$, with actions down_A from R to A , down_B from R to B , down_C from A to C , up returning to the parent, and flip negating the bit of the current room, and let $q(r, x) = r$. The context graph Γ_q has the edges $R \rightleftharpoons A$, $R \rightleftharpoons B$, and $A \rightleftharpoons C$, which is the tree of T with a return edge for every edge. The observation of a player in room r is x_r , with discrete refinement category, so the tree appears only in the navigation structure.

What each model retains.

	Filesystem	Lexical scope	Game world
role of the tree	refinement arrows $\preceq$	refinement arrows $\succeq$	navigation edges
observation at p	subtree	ancestor chain	current room
generating contexts	root	leaves	none
state-dependent access	mode bits	none	guards
δ	<code>cd</code> , <code>chmod</code> , <code>rm</code>	binding and call	movement, <code>flip</code>

The decorated tree $(T, |L|)$ with $|L| \equiv 2$ is shared by the first two and supplies the state space of the third. The translation from the filesystem to the lexical-scope system that is the identity on states is a morphism of observational reducts only after reversing the context category, and it does not preserve δ .

D Formal Inventory: Propositions, Counterexamples, and Open Questions

The inventory lists every numbered result with its level: *established* for results taken from the literature, *proved here* for results whose statement or proof is supplied in this essay, and *conjecture* or *question* for open items. The final column gives the corresponding item of the proof spine, T for theorem, P for proposition, D for definition, where the spine item was incorporated.

No.	Result	Level	Content	Spine
2.2	Prop.	established	rooted trees as posets	
2.3	Prop.	established	categories from preorders and graphs	
3.1	Prop.	proved here	no forced monotonicity of K, V, A	
3.2	Def.	definition	contextual representation system	D1
3.3	Thm.	proved here	observational refinement	T1
3.4	Prop.	proved here	independence of authority	P2
4.1	Prop.	proved here	commutation criterion	P4
4.2	Def.	definition	disclosure refinement	D2
4.3	Thm.	proved here	disclosure refinement	T3
5.1	Prop.	proved here	distinctness of <code>cd</code> , <code>subshell</code> , <code>chmod</code>	
6.1	Def.	definition	scope graph, resolution policy	D3
6.2	Thm.	proved here	determinacy of resolution	T5
6.3	Prop.	proved here	shadowing non-injective	P6
6.4	Prop.	proved here	visibility under nesting	
6.5	Prop.	proved here	scope graphs generalize trees	P7
7.1	Prop.	proved here	authority and the three orders	
8.1	Prop.	established	context graphs over-approximate	P18
8.2	Prop.	proved here	modal return not a graph edge	P18
9.3	Prop.	proved here	flat environments violate coherence	
10.1	Prop.	established	gluing criterion	T10
10.3	Prop.	proved here	disagreement excludes realization	
10.4	Ex.	proved here	obstruction to gluing	T10 target
10.5	Ex.	proved here	ambiguity, failure of separation	T10 target
11.1	Thm.	established, applied	canonical presheaf, universal property	T8
11.2	Prop.	proved here	global indistinguishability	P9
11.3	Thm.	proved here	coarsest quotient for a family of contexts	

No.	Result	Level	Content	Spine
11.4	Prop.	proved here	gluing criterion for Q	
11.5	Prop.	established	behavioral minimization	
12.1	Def.	definition	translation	D4
12.2	Prop.	proved here	composition of translations	P0
12.3	Thm.	proved here	translation preserves indistinguishability	T11
12.4	Prop.	proved here	reflection requires injectivity	P12
13.2	Thm.	proved here	common abstraction without equivalence	T13
13.3	Prop.	proved here	terminal system, trivial abstraction	
13.4	Thm.	established, applied	terminal observed system	
13.5	Cor.	proved here	what the canonical abstraction forgets	
13.6	Ex.	proved here	same abstraction, no translation back	P16
13.7	Prop.	proved here	orientation duality	P17
14.1	Thm.	established	equivalence criterion	T15
14.2	Prop.	proved here	surjectivity is not equivalence	P14
14.3	Prop.	proved here	no unrestricted faithful collapse	P20
14.4	Prop.	proved here	equivalence of posets is isomorphism	
14.5	Cor.	established	Morita rigidity for posets	
14.6	Thm.	proved here	relative equivalence	
14.8	Prop.	proved here	common quotient, no mutual translation	P16
15.1	Prop.	established	classical and intuitionistic arithmetic	
15.2	Prop.	established	Glivenko map	
15.3	Question	open	interpretations and preservation	
16.1	Prop.	established	full abstraction as canonical quotient	
16.3	Prop.	established	contexts migrate into data	
16.2	Prop.	established	universality is insufficient	P19
17.1	Thm.	proved here	partition representation	
17.2	Thm.	proved here	representation of tree systems	
17.4	Prop.	proved here	functoriality of the canonical quotient	
17.5	Question	open	pushouts of correspondences	
18.1	Prop.	proved here	reachability under translation	
18.2	Prop.	proved here	authority, prohibition, safety	
18.3	Prop.	proved here	reconstruction under translation	
18.4	Question	open	complexity of translation	
19.1	Prop.	established	unfolding is a bisimulation	
19.2	Prop.	proved here	information lost by unfolding	
19.3	Prop.	proved here	descent	
19.4	Ex.	proved here	shared room, two routes	

No.	Result	Level	Content	Spine
19.5	Conj.	conjecture	descent for dynamics	

The three levels of claim announced in Section 1 are the three values of the third column. No result classified as established is claimed to be new, and the results classified as proved here are elementary in their proofs and are offered for their formulation and their use in organizing the comparison.

E Related Frameworks and Bibliographic Notes

The categorical background is standard [21, 6], with an accessible account in [4] and the treatment of sheaves and sites in [22]. The programming-language background draws on [15, 30], program analysis on [28], and the semantics of objects on [1]. The scope-graph formulation of name resolution adopted in Section 6 is that of [29], and the use of presheaf-valued observation relative to contexts follows the spirit of sheaf-theoretic treatments of contextuality [2]. Action semantics [26] and categorical combinators [9] are alternative frameworks for modular semantics of programming languages, and the groupoid interpretation of type theory [16] is a categorical semantics in which identity is itself structured. For translation among logics, the theory of institutions [14] provides a general framework in which a logical system is a functor from signatures to satisfaction relations and translations are morphisms of such functors, and relating contextual representation systems to institutions is a natural next step noted in Section 20. For access control, the original formulations are [19, 10], the undecidability of safety is [17], and the object-capability discipline is discussed in [7]. The suggested reading list from which the present development started is a list of starting points for a literature review, and the inclusion of a work in it is not a claim that the construction proposed here has been established there.

Comparison with institutions. An institution assigns to each signature a category of models and a set of sentences, with a satisfaction relation that is invariant under change of signature [14]. Three features of the present framework are representable in that setting, and three are not without extra structure. The observational kernel $\sim_c$ of a context is the theory-indistinguishability relation of the models relative to a fixed set of sentences, so the canonical presheaf Q corresponds to the quotient of models by elementary equivalence relative to the sentences available at c , and the restriction maps correspond to signature inclusions. The terminal observed system is the set of global sections of the presheaf of sentence-evaluations. The translation conditions (O) are the satisfaction condition restricted to the observed sentences. What an institution does not carry is an order on contexts that is independent of signature inclusion, a component of authority separate from satisfaction, and a transition function on models; the lax condition (A) and the simulation condition (D) have

no counterpart there, and the failure of the commutation criterion of Section 4 is invisible to an institution because navigation is not a morphism of signatures. The framework is therefore best described as institution-like in its observational reduct and as a strict extension of it in authority and dynamics, and a precise comparison, which would exhibit the observational reduct of **CRS** as a full subcategory of a category of institutions, has not been carried out here.

Comparison with scope graphs and coalgebraic semantics. The scope-graph calculus of [29] defines name resolution by reachability under a path policy, which corresponds in this essay to the specification of the observation presheaf from a graph, and Proposition 6.5 records that scope graphs strictly generalize scope trees. The calculus does not address the comparison of resolution disciplines across languages, nor authority or transitions. Coalgebraic treatments of behavioral equivalence [37] characterize the coarsest bisimulation of a transition system, and Proposition 11.5 is the instance of this for the partial transition function used here; the observational quotient $X_{\mathcal{U}}$ differs from the behavioral quotient exactly by the distinction illustrated in Example 13.6.

Simulation-based accounts of physical reasoning. A research program in cognitive science proposes that intuitive physical judgment is produced by approximate probabilistic simulation, in the manner of a game physics engine, restricted to the objects and interactions relevant to the question posed [34]. This is an empirical hypothesis about people and is not a consequence of anything proved here, and the mathematics of this essay offers no evidence for or against it. A formal observation is nevertheless available. A simulator that instantiates only the objects relevant to a query computes, for each query, an observation at a restricted context, and the requirement that answers to related queries cohere is the coherence condition $o_c = \Omega(u) \circ o_d$. Whether the sets of queries and simulations in such an account form a presheaf satisfying the separation and gluing conditions of Section 10, and whether the failure modes of that section occur in it, are questions that the experimental literature has not posed in this form, and they are recorded as Question E.1.

Question E.1 (Simulation-based accounts as contextual systems). Can a simulation-based account of physical judgment be presented as a contextual representation system whose contexts are the sets of objects instantiated for a query, and, if so, do the observation maps satisfy separation and gluing, or do the obstruction and ambiguity modes of Section 10 occur?

References

- [1] M. Abadi and L. Cardelli, *A Theory of Objects*, Springer, 1996.
- [2] S. Abramsky and A. Brandenburger, The sheaf-theoretic structure of non-locality and contextuality, *New Journal of Physics* 13 (2011), 113036.

- [3] P. Aczel, The type theoretic interpretation of constructive set theory, in *Logic Colloquium '77*, North-Holland, 1978, 55–66.
- [4] S. Awodey, *Category Theory*, 2nd ed., Oxford University Press, 2010.
- [5] T. W. Barrett and H. Halvorson, Morita equivalence, *Review of Symbolic Logic* 9 (2016), 556–582.
- [6] F. Borceux, *Handbook of Categorical Algebra 1: Basic Category Theory*, Cambridge University Press, 1994.
- [7] M. S. Miller, K.-P. Yee, and J. Shapiro, Capability myths demolished, Technical Report SRL2003-02, Johns Hopkins University Systems Research Laboratory, 2003.
- [8] M. Carneiro, The type theory of Lean, unpublished manuscript, 2019.
- [9] P.-L. Curien, *Categorical Combinators, Sequential Algorithms, and Functional Programming*, 2nd ed., Birkhäuser, 1993.
- [10] J. B. Dennis and E. C. Van Horn, Programming semantics for multiprogrammed computations, *Communications of the ACM* 9 (1966), 143–155.
- [11] M. Felleisen, On the expressive power of programming languages, *Science of Computer Programming* 17 (1991), 35–75.
- [12] H. Friedman, Classically and intuitionistically provably recursive functions, in *Higher Set Theory*, Lecture Notes in Mathematics 669, Springer, 1978, 21–27.
- [13] V. Glivenko, Sur quelques points de la logique de M. Brouwer, *Bulletin de la Classe des Sciences, Académie Royale de Belgique* 15 (1929), 183–188.
- [14] J. A. Goguen and R. M. Burstall, Institutions: abstract model theory for specification and programming, *Journal of the ACM* 39 (1992), 95–146.
- [15] R. Harper, *Practical Foundations for Programming Languages*, 2nd ed., Cambridge University Press, 2016.
- [16] M. Hofmann and T. Streicher, The groupoid interpretation of type theory, in *Twenty-Five Years of Constructive Type Theory*, Oxford University Press, 1998, 83–111.
- [17] M. A. Harrison, W. L. Ruzzo, and J. D. Ullman, Protection in operating systems, *Communications of the ACM* 19 (1976), 461–471.
- [18] J. Lambek and P. J. Scott, *Introduction to Higher Order Categorical Logic*, Cambridge University Press, 1986.
- [19] B. W. Lampson, Protection, *Proceedings of the 5th Princeton Conference on Information Sciences and Systems*, 1971, 437–443; reprinted in *ACM Operating Systems Review* 8 (1974), 18–24.

- [20] F. W. Lawvere, Functorial semantics of algebraic theories, *Proceedings of the National Academy of Sciences* 50 (1963), 869–872.
- [21] S. Mac Lane, *Categories for the Working Mathematician*, 2nd ed., Springer, 1998.
- [22] S. Mac Lane and I. Moerdijk, *Sheaves in Geometry and Logic*, Springer, 1992.
- [23] Y. Minamide, G. Morrisett, and R. Harper, Typed closure conversion, *Proceedings of POPL 1996*, 271–283.
- [24] E. F. Moore, Gedanken-experiments on sequential machines, in *Automata Studies*, Princeton University Press, 1956, 129–153.
- [25] J. H. Morris, Lambda-calculus models of programming languages, Ph.D. thesis, Massachusetts Institute of Technology, 1968.
- [26] P. D. Mosses, *Action Semantics*, Cambridge University Press, 1992.
- [27] K. Barrett, B. Cassels, P. Haahr, D. A. Moon, K. Playford, and P. T. Withington, A monotonic superclass linearization for Dylan, *Proceedings of OOPSLA 1996*, 69–82; and M. Simionato, The Python 2.3 method resolution order, Python documentation.
- [28] F. Nielson, H. R. Nielson, and C. Hankin, *Principles of Program Analysis*, Springer, 1999.
- [29] P. Neron, A. Tolmach, E. Visser, and G. Wachsmuth, A theory of name resolution, in *Programming Languages and Systems (ESOP 2015)*, Lecture Notes in Computer Science 9032, Springer, 2015, 205–231.
- [30] B. C. Pierce, *Types and Programming Languages*, MIT Press, 2002.
- [31] G. D. Plotkin, LCF considered as a programming language, *Theoretical Computer Science* 5 (1977), 223–255.
- [32] The Open Group, *The Open Group Base Specifications, Issue 7* (IEEE Std 1003.1-2017), utility cd.
- [33] Python Software Foundation, *The Python Language Reference*, section on the execution model: naming and binding.
- [34] T. D. Ullman, E. Spelke, P. Battaglia, and J. B. Tenenbaum, Mind games: game engines as an architecture for intuitive physics, *Trends in Cognitive Sciences* 21 (2017), 649–665.
- [35] J. C. Reynolds, Definitional interpreters for higher-order programming languages, *Proceedings of the ACM Annual Conference*, 1972, 717–740.
- [36] H. Rogers Jr., Gödel numberings of partial recursive functions, *Journal of Symbolic Logic* 23 (1958), 331–341.
- [37] D. Sangiorgi, *Introduction to Bisimulation and Coinduction*, Cambridge University Press, 2011.

- [38] A. S. Troelstra and D. van Dalen, *Constructivism in Mathematics: An Introduction*, vol. 1, North-Holland, 1988.
- [39] R. Alur and P. Madhusudan, Visibly pushdown languages, *Proceedings of STOC 2004*, 202–211.