Structural Arithmetic and Continuous Approximation over the ISAR Kernel

Fabian Schindler

September 4, 2026

Abstract

We bridge the discrete combinatory logic of the ISAR substrate with continuous analysis and ML theory. We define a structural quantity model and matrix representation of carriers, showing that matrix models form a terminal matrix kernel. Universal approximation (`ISAR_UAT`) and the completion interface (`KernelAddressLimit`) remain **named axioms**: `KernelAddress` has no metric, and UAT is not proved.

Chapter 1

Introduction and Foundational Motivation

Modern programming language design, software engineering, and formal verification are plagued by a profound fragmentation. Disjoint computational models—such as the untyped lambda calculus, term rewriting systems (TRS), e-graphs, compiler intermediate representations (IR), and virtual machine bytecodes—are treated as separate worlds. They occupy distinct namespaces, employ incompatible metatheories, and rely on ad-hoc, pairwise compiler pipelines that leak implementation details and fail to preserve observational boundaries.

This paper argues that this fragmentation is unnecessary. We show that a broad class of closed formal systems can be represented through a single, quotient-mediated semantic kernel without loss of operational behavior. This kernel is not a new "universal language," but rather a view-independent substrate: different programming languages and formalisms become different *decoders* over the same invariant structure.

1.1 Primal Modeling Assumptions

To build a mathematically closed computational substrate, we establish a set of minimal design assumptions. These are not proposed as cosmological truths about reality, but rather as the foundational design requirements for a closed algebraic representation of computation.

Axiom 1 (Substrate Existence)**.** *The universe of discourse (carrier space) is non-empty:* $\Omega \neq \emptyset$.

Axiom 2 (Identity Context)**.** *The substrate contains a unique element representing the empty or inactive context, denoted by* $\emptyset$*, from which distinction begins.*

Axiom 3 (Self-Description Capability)**.** *The representation must support self-referential structures. This requires the algebraic capability to form pairs of carrier names, beginning with the initial pair:* $P = (\emptyset, \emptyset)$.

Axiom 4 (Operational Asymmetry)**.** *To drive computation, the system must distinguish the active operator from the operand it acts upon. In algebraic terms, this asymmetric relationship is modeled via Kuratowski pairing [5]:*

$$(x, y) \triangleq \{\{x\}, \{x, y\}\}$$

This asymmetry, which we denote by Δ*, serves as the operational drive for directed rewriting steps.*

1.2 The 4-Carrier System and Design Heuristics

The computational substrate is spanned by four primitive carriers: Identity (I), Rewrite (R), Adjacency (A), and State (S). We conjecture that exactly four independent roles are necessary and sufficient to form a closed, non-trivial computational substrate:

1. **Identity (C_I)**: Establishes a notion of stable normal forms and equivalence-preserving transformations. Without it, states dissolve into noise.
2. **Rewrite (C_R)**: Represents directed, non-invertible selection. Without it, transitions are symmetric, preventing one-way directed computation.
3. **State (C_S)**: Enables repeatable observables and state-space closure under self-application. Without it, the system collapses to a strictly linear, non-duplicating fragment that is sub-universal.
4. **Adjacency (C_A)**: Maps topological interaction and function application. Without it, parameters cannot be composed into ordered causal chains.

Under this design heuristic, no carrier can serve dual roles without collapsing the representation space:

- If $C_I = C_S$, the quotient structure collapses, identifying observational equivalence with syntactic identity.
- If $C_R = C_A$, causality becomes mutable, leading to non-deterministic, non-confluent rewrites.
- If $C_I = C_R$, the invariant becomes strategy-dependent, making evaluation order determine the semantic outcome.

Thus, a minimal carrier set size of exactly 4 is indicated.

Remark 1 (Sizing Hierarchies: Nibbles and Bytes)**.** *This 4-carrier structure provides a clean algebraic justification for the token hierarchies of digital computing:*

- ***1 Bit**: Distinguishes existence from absence (atom vs. void).*
- ***2 Bits**: Identifies one of the four active roles (C_I, C_R, C_A, C_S).*
- ***4 Bits (One Nibble)**: Represents a directed pair of names (an input pointer to a carrier role and an output pointer from a carrier role), expressing adjacency.*
- ***8 Bits (One Byte)**: Represents a pair of pairs—a source context and a target context—expressing a directed rewrite step.*

1.3 Occam and the Description-Length Heuristic

For a candidate substrate model M, we motivate the minimality of a 4-carrier substrate using a description-length marginal likelihood heuristic:

$$p(D \mid M) = \int p(D \mid \theta, M) p(\theta \mid M) d\theta$$

where θ represents the parameter space (matrices and rules) and D represents a reference computation. Given the 4-carrier limit, any substrate with $n > 4$ carriers introduces redundant gauge flexibility that increases description length. The terminal ISAR kernel minimizes this description length, acting as a Minimum Description Length (MDL) substrate.

Chapter 2

Continuous Approximation and Representation Space

In this chapter, we bridge the discrete combinatory logic of the ISAR substrate with continuous analysis and matrix spaces. We formalize dimensional physical quantities, matrix representations, expressive completeness, the tensor semantics, and named analytic axioms for universal approximation. Unique addresses in a completion of `KernelAddress` remain axiomatic: there is no metric on `KernelAddress`.

2.1 Physical Quantities and Matrix Representations

We define dimensional quantities and 4x4 matrix algebra.

Definition 1 (Quantities and Uncertainty)**.** *A quantity represents a physical parameter with:*

1. *A value (rational scalar).*
2. *A physical dimension (exponent exponents).*
3. *An epistemic uncertainty expression* `Uncertainty` *and* `Correlation`*.*

Covariance propagation propagates exact metric-epistemic covariance equations under addition and multiplication.

Definition 2 (Matrix4 Carriers)**.** *The concrete integer 4x4 matrix representa-*

tion used for basis combinator signatures. The primitive matrices are:

$$I_1 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \quad R_1 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix},$$
$$A_1 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \quad S_1 = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Theorem 1 (Idempotency and Nilpotency)**.** *The identity matrix I_1 is idempotent:*

$$I_1^2 = I_1$$

The core rewrite operator $K_1 = I_1 \cdot R_1 \cdot A_1 \cdot S_1$ is nilpotent:

$$K_1^2 = \mathbf{0}$$

Theorem 2 (Gauge Similarity)**.** *The two alternative matrix representations K_1 and K_2 are conjugate (similar) via the lower-triangular gauge transformation matrix P:*

$$P \cdot K_1 \cdot P^{-1} = K_2$$

proving that representation details are gauge shadows, while the terminal quotient state is invariant.

Definition 3 (Matrix View Map)**.** *The structural homomorphism mapping each ISAR term to its concrete 4x4 integer matrix signature:*

$$\mathit{term\_signature\_val} : \mathit{ITerm} \to \mathit{Matrix4}$$

Theorem 3 (Expressive Completeness)**.** *Every matrix generated by the basis combinator algebra is the image of some term:*

$$\forall M \in \mathit{ISKAlgebra}, \quad \exists t \in \mathit{ISKSubtype}, \quad \mathit{term\_signature\_val}\ t = M$$

2.2 The Categorical Matrix Bridge

We package the matrix representations as an admissible semantic kernel.

Definition 4 (Matrix Kernel)**.** *The semantic kernel MatrixKernel whose carrier is* `Matrix4` *and whose view map is* `term_signature_val`*.*

Theorem 4 (Nilpotent Collapse)**.** *Applying* `konst` *to itself maps to the zero matrix, showing the nilpotent collapse of the basis algebra:*

$$\mathit{kernelMatrixView}(\mathbf{K}\ \mathbf{K}) = \mathbf{0}$$

Theorem 5 (Matrix Terminality)**.** *Every structure-preserving morphism from* `MatrixKernel` *to* `ISAR_Kernel` *is observationally equivalent to the canonical decoding morphism.*

Theorem 6 (Unreachability of R and A)**.** *The structural homomorphism* `kernelMatrixView` *never produces the rotation matrix R_1 or adjacency matrix A_1 from a pure ISK term.*

2.3 Tensor Denotation Semantics

We bridge the discrete combinatory logic of the ISAR substrate with continuous analysis and matrix spaces by formalizing the denotational mapping of symbolic ISAR terms into an abstract rank-4 tensor space, where reductions map to extensional equality.

Definition 5 (Tensor Space Primitives)**.** *The abstract tensor space $\mathcal{T}$ is characterized by five primitive operators:*

- *Identity:* `t_norm`
- *Constant:* `t_konst`
- *Duplicator:* `t_dup`
- *Swapper:* `t_swap`
- *Composition:* `t_comp`

Definition 6 (Derived S in Tensor Space)**.** *The distributive combinator* **S** *in the tensor space is constructively derived using swapper, composition, and duplicator operators:*

$$t\_s_s \triangleq t\_app\Big(t\_app\big(t\_comp, t\_app(t\_comp, t\_dup)\big),\\ t\_app\big(t\_app(t\_swap, P), t\_norm\big)\Big)$$

where:

$$P \triangleq t\_app\big(t\_app(t\_comp, t\_comp), t\_app(t\_app(t\_comp, t\_comp), t\_swap)\big)$$

Definition 7 (Denotational Mapping)**.** *The structural denotational homomorphism mapping symbolic terms to the tensor space:*

$$denot : ITerm \to TensorSpace$$

Theorem 7 (Denotational Soundness)**.** *One-step reduction in the symbolic calculus maps to extensional equivalence of denotations:*

$$\forall t, u, \quad t \to_I u \implies denot\ t \approx_{ext} denot\ u$$

Thus, the denotational mapping factors uniquely through the Invariant Layer quotient:

$$InvariantLayer.toExtTensor : InvariantLayer \to ExtTensor$$

2.3.1 Sparse Matrix Operator Composition

In the concrete implementation of the axiomatic kernel (`scratch/isar_ski_kernel.py`), the emergent operator combinators are represented as products of the basic sparse matrices I, R, A, S:

$$\begin{aligned} \text{NORM} &\triangleq S \cdot I \\ \text{APP} &\triangleq A \cdot R \\ \text{DUP} &\triangleq S \cdot S \\ \text{COMP} &\triangleq R \cdot A \cdot S \\ \text{SWAP} &\triangleq R \cdot S \\ \text{CONST} &\triangleq I \cdot S \end{aligned}$$

2.3.2 Plex-Agent Tensor Primitives and Special Forms

Alternatively, the runtime environment and compiler stack defined in `tensor_primitives.py` specify four basic tensor primitives and four special forms derived from them:

$$\begin{aligned} \text{ATOM } (0) &\triangleq I \\ \text{PAIR } (1) &\triangleq S \\ \text{JOIN } (2) &\triangleq R \cdot A \\ \text{NORM } (3) &\triangleq I \cdot R \cdot A \cdot S \end{aligned}$$

The special forms are defined as compositions of these primitives:

$$\begin{aligned} \text{LAMBDA} &\triangleq \text{JOIN} \cdot \text{PAIR} \cdot \text{NORM} \\ \text{QUOTE} &\triangleq \text{ATOM} \cdot \text{NORM} \end{aligned}$$

along with conditional branching (`IF`) and variable bindings (`DEF`).

2.4 Continuous Universal Approximation

We formalize the continuous parameter space of the ISAR update.

Definition 8 (Non-Polynomial Activation)**.** *An activation function $\sigma \in C(\mathbb{R}, \mathbb{R})$ is non-polynomial:*

$$\forall P \in \mathit{Polynomial}\, \mathbb{R}, \quad \sigma \neq P$$

Definition 9 (Raw Address Space)**.** *The raw parameter configuration space $\mathit{RawAddress}\ d\ k$ representing a neural network configuration:*

$$\begin{aligned} \mathit{RawAddress}\ d\ k \triangleq \Sigma(N, T : \mathbb{N}), \quad & (\mathit{Fin}\ T \to \mathit{Fin}\ 4 \to \mathbb{R}) \times C(\mathbb{R}^d, \mathit{GridState}\ N) \\ & \times C(\mathit{GridState}\ N, \mathbb{R}^k) \end{aligned}$$

Definition 10 (Realization Map)**.** *The continuous function realized by a raw address configuration:*

$$\mathit{realizeRaw}\ d\ k\ \sigma : \mathit{RawAddress}\ d\ k \to C(\mathbb{R}^d, \mathbb{R}^k)$$

Definition 11 (Address Equivalence)**.** *Two raw parameter configurations are equivalent if they realize the same continuous function.*

Definition 12 (Kernel Address Space)**.** *The quotient parameter space of raw addresses modulo observational equivalence:*

$$\mathit{KernelAddress}\ d\ k\ \sigma := \mathit{Quotient}\ (\mathit{addressSetoid}\ d\ k\ \sigma)$$

We state the Universal Approximation Theorem as a **named axiom** (Leshno-style; not a theorem in this repository).

Axiom 5 (ISAR Universal Approximation)**.** *Let $K \subset \mathbb{R}^d$ be compact, $f \in C(\mathbb{R}^d, \mathbb{R}^k)$ a target continuous function, σ a non-polynomial activation, and $\varepsilon > 0$. Then there exists a raw address $\theta \in \mathit{RawAddress}\ d\ k$ such that:*

$$\forall x \in K, \quad \|\mathit{realizeRaw}\ d\ k\ \sigma\ \theta\ x - f(x)\| < \varepsilon$$

2.5 Quotient Completion and Unique Representation

We complete the parameter space only as a named interface.

Axiom 6 (Quotient Completion Space)**.** *The named placeholder type for a completion of `KernelAddress`. `KernelAddress` has no metric or uniform structure in Lean; this is **not** Mathlib `Metric.Completion`.*

Axiom 7 (Continuous Realization Limit)**.** *The named realization map on `KernelAddressLimit` (not a constructed Mathlib extension).*

Axiom 8 (Completion Embedding)**.** *Named embedding i : KernelAddress $\to$ KernelAddressLimit (not a metric-completion inclusion).*

Theorem 8 (Embedding Injectivity)**.** *The named embedding is injective (from `continuousRealization` injectivity and the commuting axiom).*

Theorem 9 (Embedding Density)**.** *Compact-open density relative to the named axiom `ISAR_UAT`, not a metric-space density theorem.*

Axiom 9 (Topological Extension Bijection)**.** *Named extension axiom. Would follow from a metric on `KernelAddress` plus density; that metric does not exist in the formalization.*

Theorem 10 (ISAR Representation Theorem)**.** *Relative to the named axioms* `ISAR_UAT` *and* `topological_extension_bijection`*: every continuous* $f \in C(\mathbb{R}^d, \mathbb{R}^k)$ *has a unique address* $\theta_{limit} \in KernelAddressLimit$ *such that:*

$$continuousRealizationLimit\ d\ k\ \sigma\ \theta_{limit} = f$$

This does not prove UAT or construct a metric completion.

Theorem 11 (Physical System Address)**.** *Wrapper:* `ISAR_representation` *applied to an arbitrary continuous map. Adds no physical content and no extra axiom.*

Chapter 3

Discussion, Related Work, and Limitations

We present a review of related literature, establish the formal boundaries of validity for our theorems, and outline directions for future research.

3.1 Related Work

The ISAR framework intersects several foundational areas of computer science and logic:

- **Combinatory Logic and Lambda Calculus**: The bracket abstraction algorithm compiled to combinator form is a classical approach pioneered by Curry and Feys [2] and Barendregt [1]. While traditional implementations suffer from code size explosion (e.g., $O(n^3)$ for standard abstraction), the linear fragment of ISAR maintains bounded sizing, sharing a direct connection to optimal reduction.
- **Optimal Evaluation and Interaction Nets**: Yves Lafont's interaction nets [6] provide a graphical formulation of computation where reductions are local and symmetric. Our formalization of the linear fragment (`LinearIKTerm`) and its bounds mirrors the behavior of optimal virtual machines like the Higher-Order Virtual Machine 2 (HVM2) [10], which compiles terms to interaction nets for parallel execution.
- **Categorical Semantics of Computation**: Lawvere's functorial semantics [7] established algebraic theories as categories. Our formulation of semantic kernels ($\mathcal{K}$) and the proof of terminality of the Invariant Layer quotient formalizes view-independence as a universal factorization property.
- **Proof Assistant Kernels**: The verification and automated generation of documentation utilizing Lean 4 quotients and metadata mappings (e.g.,

doc-gen4 [8]) represent the state-of-the-art in proof design. Our work extends this by proving formal simulation theorems directly against the proof assistant's environment.

3.2 Limitations and Scope Boundaries

The theorems presented in this work are accompanied by strict boundary conditions:

1. **Closed Dialects Only**: The terminality and factorization theorems apply exclusively to computational dialects that are *operationally closed*—meaning they possess a total encoding function into the substrate and their evaluation dynamics can be modeled as deterministic, confluent rewrite steps. This excludes open, dynamic, or referentially open systems (such as natural languages, interactive APIs, or non-deterministic transition systems) that require contextual anchors.

2. **ZFC HF Set Scope**: The set-theoretic relative interpretation is verified solely for the Hereditarily Finite set fragment. Extension to infinite sets (ZFC with the axiom of infinity) requires different ordinals and is outside the scope of the current Lean 4 proofs.

3. **Interpretive Physical Assumptions**: The application of the Universal Approximation Theorem to physical systems rests on the interpretive hypothesis that a physical system's state space can be modeled as an admissible continuous kernel. This hypothesis is a modeling choice and is not mechanically verified by the Lean 4 proof assistant.

3.3 Future Work

This paper forms the core substrate of a multi-part series of monographs:

- **Paper 2 (Continuous Approximation)**: Extension of the discrete kernel to real-valued physical quantities and continuous address spaces. Unique representation in a completion of `KernelAddress` is axiomatic (`ISAR_UAT` stays named; there is no metric on `KernelAddress`). Classical citations: Cybenko [3] and Leshno et al. [9].

- **Paper 3 (Hardware Compilation)**: Verification of the compilation path from the algebraic matrix representations to optimal virtual machines (HVM2) and relational database schemas. We also formalize partial evaluation and dialect specialization via the first, second, and third Futamura projections [4].

Bibliography

[1] Hendrik Pieter Barendregt. *The Lambda Calculus: Its Syntax and Semantics*, volume 103. Elsevier, 1984.

[2] Haskell Brooks Curry and Robert Feys. *Combinatory Logic*, volume 1. North-Holland Publishing Company, 1958.

[3] George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems*, 2(4):303–314, 1989.

[4] Yoshihiko Futamura. Partial evaluation of computation process–an approach to a compiler-compiler. *Systems, Computers, Controls*, 2(5):45–50, 1971.

[5] Kazimierz Kuratowski. Sur la notion de l'ordre dans la théorie des ensembles. *Fundamenta Mathematicae*, 2(1):161–171, 1921.

[6] Yves Lafont. Interaction nets. *Proceedings of the 17th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 95–108, 1990.

[7] F William Lawvere. *Functorial semantics of algebraic theories*. PhD thesis, Columbia University, 1963.

[8] Lean Prover Community. doc-gen4: Documentation generator for lean 4. `https://github.com/leanprover/doc-gen4`, 2023.

[9] Moshe Leshno, Vladimir Ya Lin, Allan Pinkus, and Shimon Schocken. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Networks*, 6(6):861–867, 1993.

[10] Victor Taelin. Hvm2: Higher-order virtual machine 2. `https://github.com/HigherOrderCO/HVM2`, 2024.