

Architecture of Mathematics: Internal Languages, Categorical Models, and the Spectral Lattice of Formal Systems

Groupoid Infinity

June 11, 2026

Abstract

In this article, we investigate the conjecture that no single formal language can serve effectively as a universal foundation for the entirety of mathematics, proposing instead that mathematics is best served by a system of specialized languages connected by functorial transports. We analyze the Groupoid Infinity stack of formal languages as a *lattice* of internal languages, rather than a linear tower. The languages fall into three distinct branches: a *constructive type theory and set analysis* spine (Henk–Frank–Christine), a *super homotopy* spine extending to differential cohesion and quantum spectra (Henk–Per–Anders–Urs), and an *infinity category theory* spine (Dan–Mike–Ulrik). Due to their divergent semantic and syntactic structures, attempting to unify them into a single monolithic abstract syntax tree (AST) is often impractical and conceptually strained. We demonstrate how these separate domains interact functorially — for instance, simplicial type theory (Ulrik/STT) provides the directed-homotopy backbone for Urs’s 3-layer architecture. Finally, we propose a cohomological perspective on language semantics: by associating syntactic CW-complexes and chain complexes to individual languages, one might define their cohomology groups and assemble a spectral sequence over the full lattice of language categories, outlining a conceptual mathematical framework that formalizes this ecosystem as a Grothendieck fibration of syntactic categories under Lawvere’s functorial semantics.

Contents

1	Type Systems for Working Mathematician	3
1.1	The Prerequisites	3
1.2	The Three Spines of the Language Lattice	4
1.3	The Cosmic Cube Classification	5
1.4	The Topoi Structure	6
1.5	The Semantic Foundations	7
1.6	The Internal Languages	8
2	Spine I: Set Analysis	10
2.1	Frank: Minimal Inductive System [PP89]	10
2.2	Christine: Calculus of Inductive Constructions [Pau93]	10
2.3	Laurent: Calculus and Functional Analysis [Sch50]	10
3	Spine II: Super Homotopy Type Theory	11
3.1	Henk: Pure Type Systems (PTS) [Bar92]	11
3.2	Per: Martin-Löf Type Theory (MLTT-80) [MLS84]	11
3.3	Anders: Cubical HoTT [CCHM17]	12
3.4	Urs: The Family of Layered Sublanguages	12
3.4.1	Felix (Layer I) [SS12]	13
3.4.2	Jack (Layer II) [Sch13]	13
3.4.3	Urs (Layer III) [SS22a, SS22b, SS23]	13
4	Spine III: Infinity Category Theory	14
4.1	Dan: Operational Algebra [BMSS11]	14
4.2	Mike: Directed 1-Category Type Theory [AKS15]	14
4.3	Ulrik: Simplicial ∞ -Categories [RS17, GWB25]	15
5	Spectral Sequence of Language Categories	16
5.1	Functorial Conversions and Cross-Branch Morphisms	17
5.1.1	Deep Dive: The Lift Functor (Dan $\rightarrow$ Ulrik)	18
5.1.2	Deep Dive: The Groupoid Core (Ulrik $\rightarrow$ Anders)	18
5.1.3	Case Study: Möbius Strip Representation	19
6	Cohomology of Programming Languages	19
6.1	Physical Interpretations of Cohomology Groups	20
6.2	Examples across the Spines	21
7	Conclusion	22

1 Type Systems for Working Mathematician

1.1 The Prerequisites

Definition 1.1 (Categories with Families (CwF)). A CwF consists of a category $\mathcal{C}$ of contexts and substitutions with a terminal object, equipped with a functor $Ty : \mathcal{C}^{\text{op}} \rightarrow \mathbf{Set}$, a presheaf of terms $Tm : (\int Ty)^{\text{op}} \rightarrow \mathbf{Set}$, and a context comprehension operation with projection morphisms.

Definition 1.2 (Natural Models). Following Awodey, a natural model is a category $\mathcal{C}$ with a terminal object $\mathbf{1}$ and a representable natural transformation $p : Tm \rightarrow Ty$ between presheaves. Pullbacks of p along Yoneda provide context extension and substitution.

Definition 1.3 (Grothendieck Fibration). A functor $p : \mathcal{E} \rightarrow \mathcal{B}$ is a fibration when for every $X \in \mathcal{E}$ and $u : J \rightarrow p(X)$ in $\mathcal{B}$ there exists a Cartesian lift $f : Y \rightarrow X$ in $\mathcal{E}$.

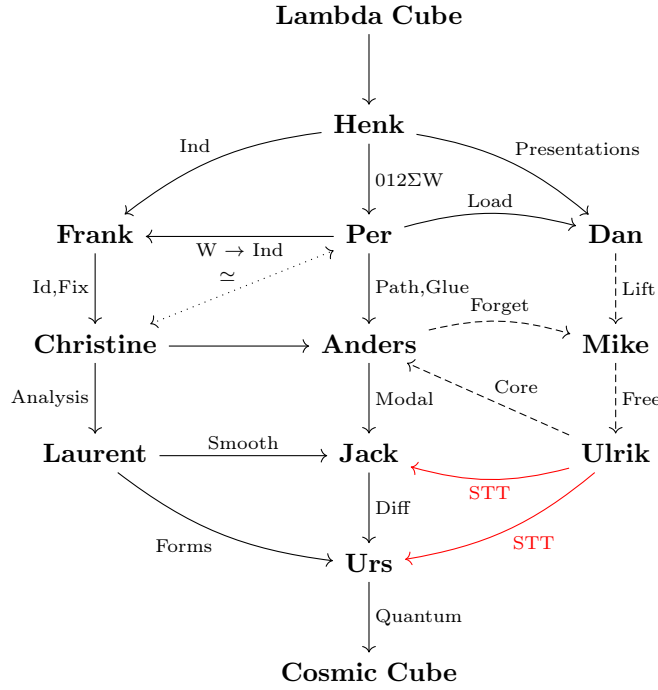
The category of languages $\mathcal{L}$ has formal languages as objects and syntactic functors (compilers, type checkers, normalizers, extractors) as morphisms.

Table 1: Semantic Categories and Core Language Primitives

System	Semantic Category / Model	Application & Physical/Mathematical Domain
O_λ	Cartesian Closed Category (CCC)	Untyped λ -calculus (evaluation and interpretation)
O_π	Symmetric Monoidal Category (SMC)	Process calculi (CCS, CSP, Milner's π -calculus)
O_μ	Tensor Category (TC)	Tensor calculus (vectorization)
Henk (O_Π)	Π-Cat (C-systems / CwF)	PTS Calculus of Constructions (functional completeness)
Frank	Π-Cat + Ind	Minimal Calculus of Inductive Constructions (strictly positive schemes)
Christine (O_Σ)	LCCC + Ind	Full CIC (MLTT-72 contextual completeness, dependent sums)
Laurent	Top/Met	Calculus and Functional Analysis (topological and metric spaces)
Per (O_W)	ΠW-Pretopos	Martin-Löf Type Theory MLTT-80 (inductive W-types)
Anders (O_I)	∞-Cat	Cubical Type Theory with Path types, no HITs (∞ -categories)
Anders+	∞-Topos	Cubical Type Theory with HITs (lightweight core, univalence)
Felix ($O_{\neg}$)	Cohesive ∞-Topos	Modal HoTT (synthetic differential supergeometry, flat/sharp)
Jack	Diff ∞-Topos	Stable homotopy (spectra, differential forms, K-theory)
Urs (O_H)	Urs ∞-Topos (Linear Types)	Quantum linear types (Hadamard operator, qubits, braids)
Dan ($O_{\triangleright}$)	Pres (Simplicial Sets / Chains)	Guarded recursion, operational algebra (face/degeneracy maps)
Mike	1-Cat	Directed 1-Category theory (directed paths, linear structures)
Ulrik	$(\infty, 1)$-Cat (Riehl-Shulman STT)	Simplicial $(\infty, 1)$ -categories (Segal, directed interval)

1.2 The Three Spines of the Language Lattice

The collection of formal internal languages does not admit a totally ordered hierarchy. Rather, the architecture factors into three principal spines. The first spine (**Frank–Christine–Laurent**) is dedicated to constructive inductive schemes and functional analysis. The central spine (**Henk–Per–Anders–Urs**) models univalent homotopy types and cohesive ∞ -topoi. The third spine (**Dan–Mike–Ulrik**) models directed synthetic category theory and discrete operational presentations. As formalized in the commutative diagram below, the respective syntaxes are mutually disjoint; all inter-language translations are strictly mediated by functorial morphisms, including left adjoint lifts, reflective embeddings, and forgetful coreflections.



Remark 1.4 (Christine $\simeq$ Per). Christine (full CIC with Id, Ind, Fix, Prop) and Per (MLTT-80 with Σ , Id, W, 0, 1, 2) are at a comparable level of expressiveness. Per serves as the foundation for Anders but can be considered interchangeable base systems (Ind for modeling general HIT as in cubicaltt). The choice between inductive schemes (Frank/Christine path) and W-types (Per path) is a design decision motivated by purity, not a strict ordering.

1.3 The Cosmic Cube Classification

The terminal object of the language lattice, the **Cosmic Cube**, classifies the various type theories by mapping the three spines to the axes of a three-dimensional coordinate system. The axes represent fundamental semantic properties of the systems:

1. **Strictness Axis** ($\Delta \leftrightarrow \nabla$): Models the distinction between strict (presheaf-like, Δ) and modal/cohesive/directed (sheaf-like, ∇) semantics.
2. **Groupoidality Axis** ($\mathbf{2} \leftrightarrow \mathbb{N}$): Distinguishes 1-categorical/set-theoretic models ($\mathbf{2}$) from higher homotopy-theoretic / ∞ -groupoid models ($\mathbb{N}$).
3. **Stability Axis** ($\mathbf{1} \leftrightarrow \mathbb{H}$): Contrasts standard cartesian logic/resource-insensitive type theory ($\mathbf{1}$) with stable/linear/quantum logic and spectra ($\mathbb{H}$).

Specifically, the three spines of the Groupoid Infinity stack correspond to moving along the axes of this Cosmic Cube:

- **Spine I (Set Analysis)**: Maps to the **Strictness Axis** ($\Delta \leftrightarrow \nabla$), exploring the transitions from strict constructive inductive systems (Frank, Christine) to modal and topological analysis (Laurent).
- **Spine II (Super Homotopy)**: Maps to the **Groupoidality Axis** ($\mathbf{2} \leftrightarrow \mathbb{N}$), moving from set-theoretic base languages (Per) to univalent homotopy type theory and ∞ -groupoids (Anders).
- **Spine III (Infinity Category Theory)**: Maps to the **Stability Axis** ($\mathbf{1} \leftrightarrow \mathbb{H}$), formalizing stable category theory, chain complexes, operational presentations, and stable spectra (Dan, Mike, Ulrik).

Under this geometric classification, the vertices of the Cosmic Cube correspond to the following configurations:

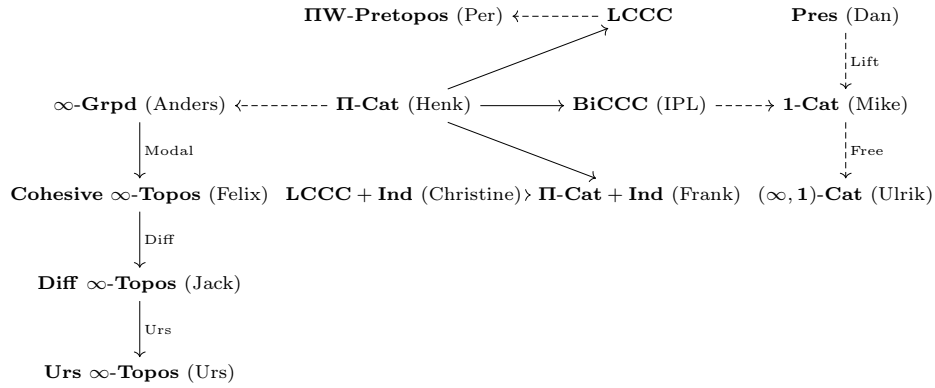
Table 2: Vertices of the Cosmic Cube

Configuration	Structure	Description & Primary System
$(\Delta, \mathbf{2}, \mathbf{1})$	$\Pi\Sigma$ (MLTT-75)	Dependently typed λ -calculus — Per
$(\Delta, \mathbf{2}, \mathbb{H})$	Linear $\Pi\Sigma$	λ -calculus, resource-sensitive computation — Urs
$(\Delta, \mathbb{N}, \mathbf{1})$	HoTT	Homotopy Type Theory — Anders
$(\Delta, \mathbb{N}, \mathbb{H})$	Linear HoTT	Linear Homotopy Type Theory — Urs
$(\nabla, \mathbf{2}, \mathbf{1})$	Modal MLTT	Modal Dependently Typed λ -calculus — Anders
$(\nabla, \mathbf{2}, \mathbb{H})$	∞ -toposes	∞ -toposes, Quantum Field Theory (dirtt, stt) — Dan
$(\nabla, \mathbb{N}, \mathbf{1})$	Modal HoTT	Synthetic Differential Geometry — Anders
$(\nabla, \mathbb{N}, \mathbb{H})$	Modal Linear HoTT	Modal Linear Homotopy Type Theory — Urs

1.4 The Topoi Structure

To analyze the gradual expansion of language primitives, we construct a fine-grained lattice of semantic categories radiating from the category of dependent products (**Π -Cat** (Henk)). The system expands in three cardinal directions (adding dependent sums to yield locally cartesian closed categories **LCCC**, inductive algebras **Π -Cat + Ind** (Frank), and bicartesian closed structures **BiCCC** (IPL)) before merging into rich categories of models such as **LCCC + Ind** (Christine), **Π W-Pretopos** (Per), **∞ -Grpd** (Anders), and cohesive **∞ -topoi**. On the right, the lattice incorporates the infinity category spine: operational algebra branch (**Pres** (Dan)), the directed 1-category branch (**1-Cat** (Mike)), and the simplicial $(\infty, 1)$ -category branch (**$(\infty, 1)$ -Cat** (Ulrik)). All cross-branch connections are detailed in the global map in Section 2.

Theorem 1.5 (Universal Primitive Lattice). *The formal internal languages form a connected lattice structure across three spines, linked by the modal, synthetic, and directed structural morphisms shown below:*



1.5 The Semantic Foundations

The lattice stratifies into structural, cohesive, linear, and directed semantics. The foundational correspondences between semantic models and their internal syntaxes progressively build up structure:

- **Structural Core:** At the base, Locally Cartesian Closed Categories (**LCCC**) provide the semantics for **MLTT**, the pure structural type theory [MLS84]. Upgrading the semantics to ∞ -categories adds globular path equality, yielding Cubical Type Theory with Path types but without Higher Inductive Types (HITs) [CCHM17]. Finally, a full ∞ -topos models Cubical Type Theory with HITs, which maintains a very lightweight core and syntax.
- **Homotopical Presheaves:** Bridging the gap from strict 1-categories to full ∞ -categories, Thierry Coquand’s Groupoid-Valued Presheaf and sheaf models (**Grpd-PSh**) [CRS21] (with higher sheaf extensions [CHS26]) provide a concrete constructive semantics for univalence, where dependent types are interpreted via the *Grothendieck Construction*. This establishes the semantic foundation for Cubical Type Theory.
- **Cohesive Modalities:** Cohesive Type Theory (Felix) [SS12] is developed specifically to model modalities such as connectedness, compactness, and infinitesimal shapes. Differential Cohesive Type Theory acts as the internal language for Differential Cohesive ∞ -topoi [Sch13], adding extra structure to ∞ -topoi just as ∞ -categories add path equality to LCCCs. These are typically modeled in the base library via undefined axioms (see `cohesivett`).
- **Linear/Tensor Modalities:** Symmetric Monoidal Categories (**SMC**) supply the semantics for resource-sensitive and directed computations. Here, Process Calculus and Interaction Nets (Lafont) act as the internal language, representing modality types with special comonadic *spawn* arrows that hide the underlying run-time implementation.
- **Directed and Simplicial Topoi:** Moving toward spatial and temporal operational algebras, presheaf categories (**Pres**) provide the semantics for Guarded Type Theory (Dan) [BMSS11]. Incorporating strict directional structures leads to **1-Cat**, acting as the semantics for Directed Type Theory (Mike) [AKS15]. Finally, scaling to higher dimensions, $(\infty, \mathbf{1})$ -**Cat** serves as the model for Simplicial Type Theory (Ulrik / Riehl-Shulman STT) [RS17, GWB25], extending directed paths with a Segal-like directed interval.

1.6 The Internal Languages

This establishes a rich hierarchy of correspondences:

$$\mathbf{LCCC} \xleftarrow{\text{Internal Language}} \mathbf{MLTT}$$

$$\mathbf{\Pi W}\text{-Pretopos} \xleftarrow{\text{Internal Language}} \mathbf{MLTT} \text{ (W – types)}$$

$$\infty\text{-Cat} \xleftarrow{\text{Internal Language}} \mathbf{Cubical} \text{ (Paths, no HITs)}$$

$$\infty\text{-Topos} \xleftarrow{\text{Internal Language}} \mathbf{Cubical} \text{ (with HITs)}$$

$$\mathbf{Cohesive} \infty\text{-Topos} \xleftarrow{\text{Internal Language}} \mathbf{Cohesive HoTT}$$

$$\mathbf{SMC} \xleftarrow{\text{Internal Language}} \mathbf{Process Calculus / Interaction Nets}$$

$$\mathbf{Pres} \xleftarrow{\text{Internal Language}} \mathbf{Guarded Type Theory} \text{ (Dan)}$$

$$\mathbf{1}\text{-Cat} \xleftarrow{\text{Internal Language}} \mathbf{Directed Type Theory} \text{ (Mike)}$$

$$(\infty, 1)\text{-Cat} \xleftarrow{\text{Internal Language}} \mathbf{Simplicial Type Theory} \text{ (Ulrik)}$$

To contextualize the development and syntactic choices of the internal languages in our stack (such as Henk, Per, Anders, Frank, and Christine), Table 3 compares the constructor support of historical and contemporary type systems. This comparison maps the evolution of dependent products (Π), dependent sums (Σ), identity types (Id), cubical paths (Ξ), universe structures (U^n), and various approaches to inductive definitions ($\mathbb{N}$, finite types $0/1/2$, W -types, general inductive definitions Ind , and general recursion/fixpoints Fix).

This breakdown highlights the design trade-offs: for instance, the Groupoid spine languages start from the minimal ECC-like core of Henk (possessing only Π and universes U^n), adding inductive types (Ind) and general fixpoints (Fix) to build Frank and Christine (CIC). In contrast, the univalent spine (Per) models

Table 3: Feature comparison of historical and contemporary type theories.

Language	U^n	Π	Σ	Id	Ξ	$\mathbb{N}$	0/1/2	W	Ind	Fix
System P_ω (CoC-88)		x								
MLTT-72		x	x							
Henk (ECC)	x	x								
Errett (LCCC/IPL)	x	x	x				x			
MLTT-73	x	x	x	x						
Per	x	x	x	x	x		x	x		
MLTT-75	x	x	x	x		x	x			
MLTT-80	x	x	x	x					x	
Anders (HTS)	x	x	x	x	x	x	x	x		
Frank (CoC+CIC)	x	x								x
Christine (Coq)	x	x							x	x
cubicaltt		x	x		x				x	x
Agda	x	x	x	x	x				x	x
Lean	x	x	x	x					x	x
NuPRL		x	x	x					x	

MLTT-80 extended with cubical path spaces (Ξ), whereas Anders (HTS) unifies these features with full cubical primitives, set-theoretic universes, and higher inductive types. Contemporary systems like Agda, Lean, and NuPRL similarly occupy different positions in this design space depending on their choice of equality (strict vs. path-based) and inductive scheme representations (general inductive types vs. W-types).

2 Spine I: Set Analysis

This spine represents the constructive/inductive spine of type theories and functional analysis. It is characterized by inductive algebraic constructions and real/functional analysis foundations.

2.1 Frank: Minimal Inductive System [PP89]

¹ Extends Henk with strictly positive inductive schemes.

```
cosmos := Uj
var := var name
pi := Π name E E | λ name E E | E E
ind := inductive name params constrs | constr i name args | ind name E args E
```

2.2 Christine: Calculus of Inductive Constructions [Pau93]

² Complete Calculus of Inductive Constructions with Σ, Id, and Ind.

```
cosmos := Prop : Uj
var := var name
pi := Π name E E | λ name E E | E E
id := Id E | ref E | idJ E
ind := inductive name params constrs | constr i name args | ind name E args E
fix := Fix
```

2.3 Laurent: Calculus and Functional Analysis [Sch50]

³ Laurent formalizes classical real and functional analysis, distribution theory, and measure theory. It provides first-class support for reals, complex numbers, measures, limits, and integration.

```
cosmos := Prop : U0 : U1
var := var ident | hole
forall := ∀ ident E E | λ ident E E | E E
exists := ∃ ident E E | (E, E) | E.1 | E.2
base := N | Z | Q | R | C | H | O | Vn
set := Set | SeqEq | And | Or | Complement | Intersect | Power | Closure | Cardinal
q := -/~ | Quot | LiftQ | IndQ
mu := mu | Measure | Lebesgue | Bochner
lim := Seq | Sup | Inf | Limit | Sum | Union
```

¹<https://frank.groupoid.space>

²<https://christine.groupoid.space>

³<https://laurent.groupoid.space>

3 Spine II: Super Homotopy Type Theory

This spine represents the central path of homotopical, cubical, and cohesive type systems, transitioning from pure dependent products to higher-dimensional geometry.

3.1 Henk: Pure Type Systems (PTS) [Bar92]

⁴ The initial object in LCC categories. Contains only Π and universes.

```
cosmos := Uj
var := var name
pi :=  $\Pi$  name E E |  $\lambda$  name E E | E E
```

3.2 Per: Martin-Löf Type Theory (MLTT-80) [MLS84]

⁵ MLTT-80 with W-types, base types 0, 1, 2, and strict equality.

```
cosmos := Uj | Vk
var := var name | hole
pi :=  $\Pi$  name E E |  $\lambda$  name E E | E E
sigma :=  $\Sigma$  name E E | (E, E) | E.1 | E.2 | E.name
id := Id E | ref E | idJ E
0 := 0 | ind0 E E E
1 := 1 | * | ind1 E E E
2 := 2 | 02 | 12 | ind2 E E E
W := W ident E E | sup E E | indW E E
```

⁴<https://henk.groupoid.space>

⁵<https://per.groupoid.space>

3.3 Anders: Cubical HoTT [CCHM17]

⁶ Semantically motivated by Coquand’s Groupoid-Valued Presheaf and sheaf models [CRS21] (where the Grothendieck construction models context extensions), Anders implements the CCHM cubical structure with Path, Glue, and higher inductive types (Coequ, Disc).

```

cosmos := Uj | Vk
var := var name | hole
pi := Π name E E | λ name E E | E E
sigma := Σ name E E | (E, E) | E.1 | E.2
0 := 0 | ind0 E E E
1 := 1 | ★ | ind1 E E E
2 := 2 | 02 | 12 | ind2 E E E
W := W ident E E | sup E E | indW E E
id := Id E | ref E | idJ E
path := Path E | Ei | E @ E
I := I | 0 | 1 | E ∨ E | E ∧ E | ¬E
part := Partial E E | [ (E = I) → E, ... ]
sub := inc E | ouc E | E [ I ↦ E ]
kan := transp E E | hcomp E
glue := Glue E | glue E | unglue E E
coequ := coequ E | ι2 | resp | indcoequ
disc := disc E | base | hub | spoke | inddisc

```

3.4 Urs: The Family of Layered Sublanguages

7

Remark 3.1 (The 3-Layer Architecture). The sublanguages represent a progressive structure for formalizing mathematical and physical concepts:

1. **Felix**: Graded universes and tensors, group actions, and super-modality operators for geometric cohesion and differential structure—bosonic/fermionic grades, flat/sharp modalities.
2. **Jack**: Homotopical foundations via spectra and groupoids, supporting differential K-theory with differential forms, connections, and refined cohomology structures.
3. **Urs**: Configuration spaces and braids for Urs statistics, supporting Urs systems and linear types (!*A*) for resource-sensitive Urs programming.

Each sublanguage culminates in a system tailored to formalize superpoints ($\mathbb{R}^{m|n}$), supersymmetry, and equivariant structures.

⁶<https://anders.groupoid.space>

⁷<https://urs.groupoid.space>

3.4.1 Felix (Layer I) [SS12]

⁸ Extends the homotopy core with graded universes, tensors, group actions, and cohesive modalities ($\flat$, $\sharp$, $\Im$) for synthetic supergeometry.

```

grade := Bose | Fermi
cosmos :=  $\mathbf{U}_{j,g}$ 
smth := SmthSet | Plot  $j$   $E$   $E$  | SupSmthSet
cohs :=  $\flat E$  |  $\sharp E$  |  $\Im E$ 
bose :=  $\bigcirc E$ 
tensor :=  $E \otimes E$ 

```

3.4.2 Jack (Layer II) [Sch13]

⁹ Extends cohesion with stable homotopy (spectra, suspensions, wedges) and differential cohomology (forms, connections, and differential K-theory).

```

grp := Grpd  $j$  | Comp  $j$   $E$   $E$   $E$ 
stable := Spectrum | Susp  $E$  |  $E \wedge E$  | HomSpec  $E$   $E$ 
ku :=  $KU_G^r E E E$  | forms  $j$   $E$  |  $d j$   $E$  | DiffKUG  $E E E E$ 

```

3.4.3 Urs (Layer III) [SS22a, SS22b, SS23]

¹⁰ The top layer, adding configuration spaces, braids, and linear resource types (!A) for Urs programming.

```

qubit := Qubit  $E E$  | app  $E E E$  | fuse  $E E E$ 
linear := !  $E$ 
topo := Config $j$   $E$  | Braid $j$   $E$ 

```

Remark 3.2 (STT $\rightarrow$ Urs: The Simplicial Connection). Simplicial type theory (Ulrik/STT) provides a direct connection to Urs. The directed interval $\mathbb{I}^\rightarrow$, extension types, and modal Π in STT supply the ∞ -categorical framework that Urs’s Layer I (Homotopy) builds upon. By isolating the category of categories **Cat** as a reflective subuniverse in simplicial type theory, STT provides the required directed homotopy structures. Concretely, STT’s directed interval becomes the path/homotopy backbone, while its modal/shape operations inform Urs’s cohesive modalities ($\flat$, $\sharp$, $\Im$) in Layer II. This is represented by the STT arrow from Ulrik to Urs in the lattice diagram, which is *not* simply a factoring through Anders.

⁸<https://felix.groupoid.space>

⁹<https://jack.groupoid.space>

¹⁰<https://urs.groupoid.space>

4 Spine III: Infinity Category Theory

This spine encompasses operational algebra and directed category theories, dealing with presentations of algebraic shapes and directed categorical structures.

4.1 Dan: Operational Algebra [BMSS11]

¹¹ Dan operates on a completely different level: it manipulates algebraic presentations (groups, simplicial sets, chain complexes) operationally. Its term language has no dependent types, no universes, and no Π/Σ . Instead it works with group words, permutations, matrices, face/degeneracy maps, and boundary operators.

```

type := Simplex | Group | Simplicial | Chain | Category
      Monoid | Ring | Field | PermGroup | MatGroup
      FpGroup | PcGroup | Module | Set | GSet
term := name |  $E \circ E$  |  $E^{-1}$  |  $E^j$  | e | matrix args
       $E + E$  |  $E \cdot E$  | perm args | cycle args
      PermGroup args | MatGroup args j k
      FpGroup names args | PcGroup names args
       $d_j E$  |  $s_j E$  |  $\partial E$  |  $E \cdot E$  | Hom  $E E$  |  $E \equiv E$ 
hypo := decl names type | eq name  $E E$  | map name  $E E$ 
      pres name  $E$  | rel  $E$  | act name  $E E$ 
      orbit  $E E$  args | stabilizer  $E E E$ 

```

4.2 Mike: Directed 1-Category Type Theory [AKS15]

¹² A *linear* type theory with a bidirectional type checker enforcing quadraticality (each category variable occurs exactly once covariantly and once contravariantly).

```

var := var name | name args
op := op  $E$ 
prod :=  $E \times E$ 
hom := Hom $E$   $E E$ 
tensor :=  $E \otimes E$ 
pi :=  $E \multimap E$  |  $\lambda$  name  $E$  |  $E E$ 
coend :=  $\int^E$  name  $E$  | coend name name name  $E$ 
end :=  $\int_E$  name  $E$  | end name  $E$ 
id := id  $E$ 
idJ := id $J$   $E$  | id $J, cov$   $E$  | id $J, contra$   $E$ 

```

¹¹<https://dan.groupoid.space>

¹²<https://mike.groupoid.space>

4.3 Ulrik: Simplicial ∞ -Categories [RS17, GWB25]

¹³ Extends Mike’s foundation with directed interval $\mathbb{I}$, shape inclusions ($\varphi \subseteq \psi$), extension types, modal Π , and twisted arrow categories (A^{tw}).

To construct a type theory for synthetic category theory, one might hope to interpret type theory in the category of categories (∞ -categories or otherwise) to ensure that types realize categories. However, the category of small categories is too poorly behaved to form a model of Martin-Löf Type Theory (MLTT). Instead, Riehl and Shulman (2017) [RS17] use the category of bisimplicial sets (simplicial spaces) to model synthetic $(\infty, 1)$ -categories via Rezk spaces, enabling further formalizations such as the Yoneda lemma and presheaf categories [GWB25]. Simplicial Type Theory (STT) axiomatizes this framework using a directed interval and shapes, allowing one to isolate $(\infty, 1)$ -categories as a reflective subuniverse of types within the theory.

```

cosmos := U
var := var name
pi :=  $\Pi$  name E E |  $\lambda$  name E E | E E
sigma :=  $\Sigma$  name E E | (E, E) | E.1 | E.2
id := Id E E E | ref E
idir :=  $\mathbb{I}$  | 0 | 1 | E  $\leq$  E |  $\Phi \subseteq \Psi$ 
part := Partial E E | [ (E = I)  $\rightarrow$  E, ... ]
ext := Ext E E E
modalpi :=  $\Pi$  shape E E |  $\lambda$  shape E | E E
tw := Tw E | twp0 E | twp1 E
ops := E  $\vee$  E | E  $\wedge$  E |  $\neg$ E
idJ := idJ E | idJ,cov E | idJ,contra E

```

¹³<https://ulrik.groupoid.space>

5 Spectral Sequence of Language Categories

Unlike a simple linear filtration, the language category $\mathcal{L}$ is organized as a *lattice* (a partially ordered set $\mathcal{I}$) with multiple join-paths. To compute the cohomology of the total language $\mathcal{L}$ from its sublanguages, we can employ the Bousfield-Kan spectral sequence for the homotopy colimit over the poset category $\mathcal{I}$:

$$E_2^{p,q} = H^p(\mathcal{I}; \mathcal{H}^q) \implies H^{p+q}(\mathcal{L})$$

where $\mathcal{H}^q : \mathcal{I} \rightarrow \mathbf{Ab}$ is the functor mapping each language $\mathcal{O}$ to its cohomology $H^q(\mathcal{O})$, and $H^p(\mathcal{I}; -)$ denotes the cohomology of the category $\mathcal{I}$.

Alternatively, one can impose an integer rank filtration on the syntactic complex $C^*(\mathcal{L})$ by defining the depth p of a language in the lattice. Let $F_p C^*(\mathcal{L})$ be the subcomplex generated by all languages of depth $\leq p$. The associated spectral sequence is:

$$E_1^{p,q} = H^{p+q}(F_p/F_{p-1}) \implies H^{p+q}(\mathcal{L})$$

where the relative complex F_p/F_{p-1} isolates the “new” syntactic primitives introduced exactly at depth p .

p	Language	Branch	Primitives added	$ C_0 $
0	Henk	Set Theory	Π, U^n	5
1	Frank	Set Theory	+ Ind	8
2a	Christine	Set Theory	+ Id, Fix	14
2b	Per	Set Theory	+ Σ , Id, W, 0, 1, 2	26
3	Anders	Homotopy Theory	+ Path, I, Glue, HComp, HIT	40+
4a	Felix	Homotopy Theory	+ $\flat$, $\sharp$, $\Im$, $\otimes$, Plot	10
4b	Jack	Homotopy Theory	+ Spectra, Susp, Grpd, Forms, DiffKU	12
4c	Urs	Homotopy Theory	+ Qubit, Linear, Config, Braid	7
—	Dan	Category Theory	Groups, Simplices, Chains, Boundaries	20
—	Mike	Category Theory	Hom, $\otimes$, $\int$, $\prod$, $\multimap$	13
—	Ulrik	Category Theory	+ $\mathbb{I}^{\rightarrow}$, Ext, Tw, $\leq$	25

Table 4: The language lattice: filtration indices and constructor counts. Note the split at $p = 2$ (Christine/Per), the four sublanguages of Urs at $p = 4$ (Homotopy, SupSmth, DiffK, Urs), and the separate branch indices for Dan, Mike, and Ulrik.

Definition 5.1 (Morphisms in $\mathcal{L}$). Morphisms $f : \mathcal{O}_x \rightarrow \mathcal{O}_y$ are classified as:

1. $\mathcal{F}_{\text{enrich}}$: Enrichment (type inference, elaboration).
2. $\mathcal{F}_{\text{simp}}$: Simplification (type erasure, extraction).
3. $\mathcal{F}_{\text{trans}}$: Translation (compilation across branches).
4. $\mathcal{F}_{\text{eval}}$: Evaluation (normalization to values).
5. $\mathcal{F}_{\text{norm}}$: Normalization ($\beta\eta$ -reduction).
6. $\mathcal{F}_{\text{certify}}$: Certification (proof checking).

Cross-branch morphisms (e.g. **Lift**: Dan $\rightarrow$ Mike, **Core**: Ulrik $\rightarrow$ Anders) are structure-changing and do not preserve the term type.

5.1 Functorial Conversions and Cross-Branch Morphisms

The language category $\mathcal{L}$ is not merely a set of objects; the relationships between the branches are mediated by structured functorial conversions. The primary conversions, their statuses, and associated information losses are detailed in Table 5.

Functor	Source	Target	Status	Semantics & Information Loss
Lift	Dan	Mike	Yes	Semantical lifting of presentation generators and equations to 1-categorical structures.
Forget	Mike	Dan	Partial	Truncates categories back to operation-based presentation generators.
Free	Mike	Ulrik	Yes	Fully faithful embedding of directed 1-categories into directed ∞ -categories.
Trunc	Ulrik	Mike	No	Cannot be done functorially without losing higher-dimensional coherence.
Core	Ulrik	Anders	Yes	Groupoid Core: discards directed path structure, retaining only the sub-groupoid of isomorphisms.
Disc	Anders	Ulrik	Yes	Discrete embedding: interprets ∞ -groupoids as discrete ∞ -categories via disc modality.
Forget	Anders	Mike	Partial	1-category approximation: projects higher paths onto 1-category hom-sets.
Groupoid	Mike	Anders	Yes	Forgetful functor from directed 1-categories to spaces of isomorphism paths.
Lift (Composite)	Dan	Ulrik	Yes	Composition of Lift and Free : maps operations to Rezk ∞ -categories.
Forget	Ulrik	Dan	No	Directed simplicial spaces are too rich to map back to discrete operations directly.
Forget	Anders	Dan	No	Undirected spaces cannot be functorially mapped to discrete presentation equations.
Modal	Anders	Felix	Yes	Extends cubical HoTT with cohesive modalities ($\flat$, $\sharp$, $\Im$) and super-grades.
Diff	Felix	Jack	Yes	Stable extension to spectra, suspensions, forms, and differential K-theory.
Quantum	Jack	Urs	Yes	Extends differential K-theory with configuration spaces, braids, and linear types.

Table 5: Primary conversions and functors between the language layers.

5.1.1 Deep Dive: The Lift Functor ($\mathbf{Dan} \rightarrow \mathbf{Ulrik}$)

The **Lift** functor acts as a semantical compilation bridge that transports combinatorial presentations from the operational layer **Dan** to the directed synthetic type theories of **Mike** and **Ulrik**. Rather than a simple syntax translator, it elaborates discrete presentations into rich categorical models in four distinct phases:

1. **Signature Ingestion:** Reads the generator set, face maps (∂_i) , degeneracy maps, and algebraic presentation relations directly from the **Dan** syntax.
2. **Semantical Lifting:** Maps the ingested signature into category-theoretic structures, translating generator equations into functors, categories, and sheaves.
3. **Elaboration to STT:** Translates composition equations to identity paths ($\mathbf{EId}$) in **Ulrik**. The directed interval $\mathbb{I}^\rightarrow$ is employed to model the domains and boundary conditions.
4. **Metatheory Absorption:** Automatically equips the lifted object with the abstract category-theoretic library of **Ulrik** (including Yoneda lemmas, limits, adjunctions, and Kan extensions).

5.1.2 Deep Dive: The Groupoid Core ($\mathbf{Ulrik} \rightarrow \mathbf{Anders}$)

The translation from **Ulrik** to **Anders** is a structural forgetful conversion that extracts the underlying homotopy type (the ∞ -groupoid of isomorphisms) from a directed ∞ -category. Since the simplicial directed interval $\mathbb{I}^\rightarrow$ and directed hom-types $(x \rightarrow y)$ of **Ulrik** do not exist in the cubical undirected world of **Anders**, the functor operates as follows:

- Discards directed arrows, keeping only the sub-groupoid A^{core} containing morphisms with valid inverses.
- Discards the Segal and Rezk completeness conditions, transforming directed category composition into symmetric path composition.
- Replaces the directed interval $\mathbb{I}^\rightarrow$ with the cubical symmetric interval I (with connection and reversal operations).

5.1.3 Case Study: Möbius Strip Representation

To illustrate the conversions, Table 6 compares the representation of a Möbius Strip across the three primary layers (**Dan**, **Ulrik**, and **Anders**).

Table 6: Comparison of the Möbius Strip representation across the layers.

Aspect	Dan (Operational)	Ulrik (Simplicial STT)	Anders (Cubical HoTT)
Framework	Algebra presentations	STT with directed interval $\mathbb{I} \rightarrow$	Cubical HoTT with modalities
Math Type	Simplicial complex with boundaries	Rezk ∞ -category from simplices	Bundle $\sum_{x:S^1} \text{Moebius}(x)$
Twist Encoding	Equations on face maps (∂_i)	Directed paths satisfying Segal	Path transport (ua) via not
Homotopy Type	Contractible local complex ($*$)	∞ -category with core $\simeq S^1$	Homotopy equivalence to S^1
Abstraction	Combinatorial building blocks	Synthetic directed ∞ -category	Synthetic univalent bundle
Functor Action	Ingested by Lift	Lifted to directed ∞ -category	Extracted via Core
Information	Combinatorial generators	Full directed structure	Undirected homotopy skeleton

6 Cohomology of Programming Languages

Each language’s syntax is analyzed *individually* as a CW-complex.

For a formal language $\mathcal{O}_x$, we associate a chain complex of abelian groups (free $\mathbb{Z}$ -modules) $C_*(\mathcal{O}_x)$ constructed from the syntax and reduction/conversion semantics. This construction models the syntactic space as a cell complex (CW-complex) where paths and higher homotopies encode the computational rules.

Definition 6.1 (Syntactic CW-Complex). For a language $\mathcal{O}_x$ with term syntax T generated by a signature Σ , the syntactic cell complex $C_*(\mathcal{O}_x)$ is defined in low dimensions by:

- **0-cells** ($C_0(\mathcal{O}_x)$) : The free $\mathbb{Z}$ -module generated by the set of basic syntactic constructors $c \in \Sigma$ (e.g., **Var**, **Pi**, **Lam**, **App** for Henk). Any term $t \in T$ has an associated constructor-occurrence representation:

$$[t] = \sum_{c \in \Sigma} \text{count}(c, t) \cdot c \in C_0(\mathcal{O}_x)$$

where $\text{count}(c, t)$ is the number of times the constructor c appears in the syntax tree of t .

- **1-cells** ($C_1(\mathcal{O}_x)$): The free $\mathbb{Z}$ -module generated by the rewriting and reduction rules $r \in \mathcal{R}$ of the language (e.g., β -reductions, evaluation steps). Each 1-cell $r : t \rightarrow t'$ represents a directed rewrite path from a redex term t to a contractum term t' .
- **2-cells** ($C_2(\mathcal{O}_x)$): The free $\mathbb{Z}$ -module generated by equivalence relations, critical pairs, confluence diagrams, and conversion equations (e.g., η -conversions, commuting conversions). A 2-cell e represents a coherence condition or a closed loop of rewrite steps.

- **Higher cells** ($C_k(\mathcal{O}_x)$ for $k \geq 3$): Coherence relations between equations, such as Mac Lane pentagon relations, path-type univalence coherences (in Anders), or quadraticity constraints (in Mike).

Definition 6.2 (Boundary Operators). The boundary operators $\partial_k : C_k(\mathcal{O}_x) \rightarrow C_{k-1}(\mathcal{O}_x)$ are defined as follows:

1. The 1-boundary operator $\partial_1 : C_1(\mathcal{O}_x) \rightarrow C_0(\mathcal{O}_x)$ is the linear map determined by the net change in constructor counts caused by a rewrite rule $r : t \rightarrow t'$:

$$\partial_1(r) = [t'] - [t]$$

2. The 2-boundary operator $\partial_2 : C_2(\mathcal{O}_x) \rightarrow C_1(\mathcal{O}_x)$ is determined by the boundary of confluence diagrams or conversion loops. If a 2-cell e represents a confluence diagram from a term s reducing to s' via two different paths of rewrites $p_1 = (r_{1,1}, \dots, r_{1,m})$ and $p_2 = (r_{2,1}, \dots, r_{2,n})$, the boundary is:

$$\partial_2(e) = \sum_{j=1}^m r_{1,j} - \sum_{k=1}^n r_{2,k}$$

Since both paths p_1 and p_2 start at s and end at s' , the net constructor changes must match:

$$\partial_1(p_1) = [s'] - [s] = \partial_1(p_2)$$

Thus:

$$\partial_1(\partial_2(e)) = \partial_1(p_1) - \partial_1(p_2) = 0$$

This proves that $\partial_1 \circ \partial_2 = 0$, establishing a valid chain complex.

Definition 6.3 (Syntactic Cohomology). Let $C_*(\mathcal{O}_x)$ be the syntactic chain complex of $\mathcal{O}_x$ and let G be an abelian group of coefficients. The cochain complex $C^*(\mathcal{O}_x; G) = \text{Hom}(C_*(\mathcal{O}_x), G)$ has coboundary operators $\delta^k : C^k(\mathcal{O}_x; G) \rightarrow C^{k+1}(\mathcal{O}_x; G)$ defined by $\delta^k(\phi) = \phi \circ \partial_{k+1}$. The syntactic cohomology groups are:

$$H^k(\mathcal{O}_x; G) = \ker(\delta^k) / \text{im}(\delta^{k-1})$$

6.1 Physical Interpretations of Cohomology Groups

The cohomology groups $H^k(\mathcal{O}_x; G)$ measure fundamental properties of the syntax and semantics of $\mathcal{O}_x$:

- $H^0(\mathcal{O}_x; G)$: Measures the *syntactic dimension* or independent constructor classes. A class in H^0 is a function $f : C_0 \rightarrow G$ such that for any rewrite $r : t \rightarrow t'$, $f([t]) = f([t'])$. Thus, H^0 captures invariants preserved under computation (such as type preservation or conserved weights of terms).

- $H^1(\mathcal{O}_x; G)$: Measures *normalization obstructions*. An element in H^1 is represented by a 1-cocycle (an assignment of weights to rewrites that sums to zero on closed loops) modulo those coming from constructor valuations. Non-trivial H^1 classes indicate non-confluent rewrites, cycles of reductions (non-termination), or irreversible compilation steps.
- $H^2(\mathcal{O}_x; G)$: Measures *coherence obstructions*. It captures the obstructions to filling confluence diagrams with higher-dimensional conversion cells. A non-trivial class in H^2 represents a critical pair (or confluence loop) that cannot be closed by conversions or equivalence cells, pointing to deep semantic mismatches or failures of univalence/coherence.

6.2 Examples across the Spines

Example 6.4 (Spines I/II: Strict & Homotopical).

- **Henk** ($\mathcal{O}_\Pi$): The core type theory.

$$C_0 = \{\mathbf{Var}, \mathbf{Universe}, \mathbf{Pi}, \mathbf{Lam}, \mathbf{App}\}$$

The 1-cells are generated by β -reduction:

$$r_\beta : \mathbf{App}(\mathbf{Lam}(x, A, t), u) \rightarrow t[u/x]$$

$\partial_1(r_\beta) = [t[u/x]] - (\mathbf{App} + \mathbf{Lam} + [t] + [u])$. The 2-cells consist of η -conversions:

$$e_\eta : \mathbf{Lam}(x, A, \mathbf{App}(t, x)) \equiv_\eta t$$

Since Henk is strongly normalizing and confluent (Church-Rosser), the higher cohomology $H^k(\mathcal{O}_\Pi; \mathbb{Z})$ for $k \geq 1$ is trivial, reflecting a contractible syntactic space.

- **Anders** ($\mathcal{O}_{\text{Anders}}$): Adds univalence and Glue types. The univalence axiom represents a path-space equivalence between isomorphic types. This introduces higher-dimensional coherences (Glue cells) acting as 3-cells, preventing non-trivial 2-dimensional obstructions.
- **Urs** ($\mathcal{O}_{\text{Urs}}$): Adds modalities ($\flat, \sharp$) and differential structures (forms, connections). The syntactic CW-complex of $\mathcal{O}_{\text{Urs}}$ is enriched with topological and differential cells, yielding non-trivial H^k corresponding to de Rham cohomology and Chern-Weil invariants of smooth/differential type structures.

Example 6.5 (Spine III: Category Theory).

- **Dan** ($\mathcal{O}_{\text{Dan}}$): Operational algebra. Here, term syntax encodes algebraic operations (e.g., face maps ∂_i , degeneracies s_i). The cell complex $C_*(\mathcal{O}_{\text{Dan}})$ is isomorphic to the standard simplicial chain complex of the underlying category/algebraic presentation.

- **Mike** ($\mathcal{O}_{\text{Mike}}$): Directed category theory. 1-cells are non-reversible functors/morphisms, meaning paths are directed. The boundary operator ∂_1 tracks directed change. Quadraticity constraints act as 2-cells that model the non-commutative relation profiles of monoidal/directed types.
- **Ulrik** ($\mathcal{O}_{\text{Ulrik}}$): Simplicial ∞ -categories. Generalizes Mike’s 1-categories to directed simplicial spaces. The higher cells (C_k for $k \geq 2$) represent the Segal completeness conditions and Rezk completeness conditions, ensuring that the directed paths compose coherently up to higher homotopies.

7 Conclusion

A recurring ambition in the development of higher-dimensional type theories is the pursuit of a singular, comprehensive foundational language — a formal system capable of expressing discrete combinatorial presentations, directed synthetic category theory, and univalent cohesive geometry simultaneously. However, as demonstrated by the categorical obstructions documented in this article, we conclude that *no natural unified foundation is viable*. Any attempt to internalize this entire hierarchy into a single categorical semantics would inevitably require rigidification into strict bicategories, tricategories, or infinitely coherent higher-categorical structures. Such a global strictification is highly non-natural and introduces prohibitive complexity. Instead, the mathematical reality of higher-dimensional formalisms naturally forms a *functorial mesh of transports*.

The obstructions to a unified foundation are deeply rooted in the underlying geometry of the respective topoi. For example, consider the embedding of Simplicial Type Theory (**Ulrik**) into Cubical HoTT (**Anders**). The directed interval $\mathbb{I} \rightarrow$ and its associated Segal and Rezk completeness conditions depend fundamentally on an asymmetrical, directed path space. When attempting to interpret this within the symmetric, invertible cubical paths of **Anders**, one is forced to invoke the **Core** functor. This functor aggressively truncates the directed ∞ -category, extracting only its maximal ∞ -groupoid of isomorphisms. The directed geometric structure is irreversibly collapsed by this essential geometric morphism.

Similarly, the transition from discrete algebraic presentations (**Dan**) to synthetic homotopy theories cannot be accomplished via internal deformation. The combinatorial algebra of face maps, degeneracy maps, and strict boundary operators resides in a category devoid of continuous path spaces or univalent universes. To interpret these discrete generators within the synthetic structures of **Mike** or **Ulrik**, one must employ the **Lift** functor to fully realize them as categorical objects. There exists no continuous adjunction from this rigid, discrete combinatorial layer into the cohesive ∞ -topoi of **Felix** or the linear configuration spaces of **Urs**.

Ultimately, the architecture of modern higher-dimensional mathematics is irreducibly pluralistic. The “Groupoid Infinity” stack acknowledges this by treating specialized theories not as fragments of a missing monolith, but as distinct

objects in a category of formal languages. By relying on a functorial mesh of geometric morphisms, discrete embeddings, and modal coreflections, we achieve a robust and mathematically rigorous ecosystem whose compositional power far exceeds that of any artificially strictified unified theory.

References

- [AKS15] Benedikt Ahrens, Krzysztof Kapulkin, and Michael Shulman. Univalent categories and the rezk completion. In Maria del Mar González, Paul C. Yang, Nicola Gambino, and Joachim Kock, editors, *Extended Abstracts Fall 2013*, pages 75–76, Cham, 2015. Springer International Publishing.
- [Bar92] H. P. Barendregt. Lambda calculi with types. In S. Abramsky, Dov M. Gabbay, and S. E. Maibaum, editors, *Handbook of Logic in Computer Science (Vol. 2)*, pages 117–309, New York, NY, USA, 1992. Oxford University Press, Inc.
- [BMSS11] Lars Birkedal, Rasmus Ejlers Mogelberg, Jan Schwinghammer, and Kristian Stovring. First steps in synthetic guarded domain theory: step-indexing in the topos of trees. In *2011 IEEE 26th Annual Symposium on Logic in Computer Science*, pages 55–64. IEEE, 2011.
- [CCHM17] Cyril Cohen, Thierry Coquand, Simon Huber, and Anders Mörtberg. Cubical type theory: A constructive interpretation of the univalence axiom. *FLAP*, 4(10):3127–3170, 2017.
- [CHS26] Thierry Coquand, Jonas Höfer, and Christian Sattler. Constructive higher sheaf models with applications to synthetic mathematics. Extended version, <https://arxiv.org/pdf/2605.15126>, 2026.
- [CRS21] Thierry Coquand, Fabian Ruch, and Christian Sattler. Constructive sheaf models of type theory. *Mathematical Structures in Computer Science*, 31(9):979–1002, 2021.
- [Sch50] Laurent Schwartz. *Théorie des distributions*. Hermann, Paris, 1950.
- [MLS84] P. Martin-Löf and G. Sambin. *Intuitionistic type theory*. Studies in proof theory. Bibliopolis, 1984.
- [Pau93] Christine Paulin-Mohring. Inductive definitions in the system Coq — rules and properties. In *Typed Lambda Calculi and Applications, International Conference on Typed Lambda Calculi and Applications, TLCA '93, Utrecht, The Netherlands, March 16-18, 1993, Proceedings*, pages 328–345, 1993.

- [PP89] Frank Pfenning and Christine Paulin-Mohring. Inductively defined types in the calculus of constructions. In *Mathematical Foundations of Programming Semantics, 5th International Conference, Tulane University, New Orleans, Louisiana, USA, March 29 - April 1, 1989, Proceedings*, pages 209–228, 1989.
- [GWB25] Daniel Gratzer, Jonathan Weinberger, and Ulrik Buchholtz. The Yoneda embedding in simplicial type theory. Ukrainian Translation: <https://groupoid.space/friends/ulrik/yoneda-simplicialtt-ua.pdf>, 2025.
- [RS17] Emily Riehl and Michael Shulman. A type theory for synthetic ∞ -categories. *arXiv preprint arXiv:1705.07442*, 2017.
- [Sch13] Urs Schreiber. Differential cohomology in a cohesive infinity-topos. 2013.
- [SS12] Urs Schreiber and Michael Shulman. Quantum gauge field theory in cohesive homotopy type theory. In *Workshop on Quantum Physics and Logic*, 2012.
- [SS22a] Hisham Sati and Urs Schreiber. Anyonic topological order in twisted equivariant differential (TED) K-theory. *arXiv preprint arXiv:2206.13563*, 2022.
- [SS22b] Hisham Sati and Urs Schreiber. Topological quantum programming in TED-K. *arXiv preprint arXiv:2209.08331*, 2022.
- [SS23] Hisham Sati and Urs Schreiber. Anyonic defect branes and conformal blocks in twisted equivariant differential (TED) K-theory. *Reviews in Mathematical Physics*, 35(08):2350024, 2023.