Spherepop: A Language for Geometric Computation

Unified Rigorous Specification

Flyxion

October 24, 2025

Contents

1 Preliminaries **1**
1.1 Syntax of Types and Terms . . . 1
1.2 Contexts, Universes, and Definitional Equality . . . 1
1.3 Substitution and Weakening . . . 2
1.4 Free Variables and Alpha-Equivalence . . . 2
1.5 Context Validity . . . 2

2 Type System **2**
2.1 Judgment Forms . . . 2
2.2 Structural Rules . . . 2
2.3 Definitional Equality Rules . . . 3
2.4 Dependent Product Π . . . 3
2.5 Dependent Pair Σ . . . 3
2.6 Dependent Pair Projections . . . 3
2.7 Products and Functions . . . 4
2.8 Distribution Type $\mathsf{Dist}(A)$ . . . 4

3 Operational Semantics **4**
3.1 Values and Contexts . . . 4
3.2 Deterministic Fragment . . . 4
3.3 Merge Operational Semantics (Complete) . . . 4
3.4 Stochastic Fragment . . . 5
3.5 Probability Measures . . . 5

4 Metatheory **6**
4.1 Confluence (Deterministic Fragment) . . . 7
4.2 Strong Normalization (Restricted Fragment) . . . 7
4.3 Decidability and Undecidability . . . 7

5 Geometric Semantics **7**
5.1 RSVP Model . . . 7
5.2 Information-Theoretic Compression . . . 8

6 DSL-to-Core Translation **9**
6.1 Lexical Syntax 9
6.2 Context-Free Grammar (EBNF) 9
6.3 Operator Precedence 10
6.4 Translation Semantics 10

7 Recursion and `spin` **10**

8 Category-Theoretic Semantics **11**
8.1 Base Category 11
8.2 Cartesian Closed Structure 12
8.3 Monoidal Structure of Merge 12
8.4 Probabilistic Monad 12
8.5 Presheaf Topos Model 13

9 Implementation Roadmap **13**
9.1 Module Structure 13
9.2 Key Algorithms 14
9.2.1 Normalization by Evaluation (NbE) 14
9.2.2 Bidirectional Type Checking 14

10 Worked Example (End-to-End) **14**

11 Extended Examples **15**
11.1 Example 1: Identity Function 15
11.2 Example 2: Composition with Merge 15

12 Spherepop II: Derived Geometric Semantics and Shifted Symplectic Structure **16**
12.1 Differential Operator 16
12.2 Shifted Symplectic Form 16
12.3 Derived Category **Geom** 16

Abstract

Spherepop is a programming language and formal calculus that treats computation as geometry. This unified specification integrates a human-facing geometric DSL, a typed core calculus (SPC), probabilistic operational semantics, and a rigorous geometric interpretation under the RSVP model. It includes complete typing rules, metatheoretic proofs, DSL translation, and categorical coherence.

1 Preliminaries

Natural-language explanation. This section establishes notation, contexts, universes, substitution, definitional equality, free variables, and context validity.

1.1 Syntax of Types and Terms

Definition 1 (Types and Terms)**.**

$$A, B ::= \mathcal{U}_i \mid \Pi x : A.\, B \mid \Sigma x : A.\, B \mid A \to B \mid A \times B \mid \mathsf{Dist}(A)$$
$$t, u, v ::= x \mid a \mid \mathsf{Sphere}(x : A.\, t) \mid \mathsf{Pop}(t, u) \mid \mathsf{Merge}(t, u) \mid \mathsf{Choice}(p, t, u) \mid \langle t, u \rangle$$

Explanation. Types form a cumulative universe hierarchy $\mathcal{U}_0 \in \mathcal{U}_1 \in \mathcal{U}_2 \cdots$. Core constructors support dependent functions, pairs, non-dependent arrows and products (derivable), and distributions for stochastic computation. Terms explicitly include dependent pairs.

1.2 Contexts, Universes, and Definitional Equality

Definition 2 (Contexts and Universes)**.** A context Γ is a list $x_1 : A_1, \ldots, x_n : A_n$ with distinct variables. Universe rules:

$$\Gamma \vdash \mathcal{U}_i : \mathcal{U}_{i+1}, \qquad i < j \Rightarrow \Gamma \vdash \mathcal{U}_i : \mathcal{U}_j.$$

Definition 3 (Definitional Equality)**.** $t \equiv u$ and $A \equiv B$ form the least congruence containing β- and η-rules for Π, extended to Σ, products, and primitives. We write $\Gamma \vdash t \equiv u : A$.

Explanation. An intensional dependent type theory core ensures conversion via definitional equality.

1.3 Substitution and Weakening

Definition 4 (Simultaneous Substitution)**.** Define $t[\vec{u}/\vec{x}]$ inductively on term structure...

Lemma 1 (Substitution Composition)**.** $(t[u/x])[v/y] = t[v/y][u[v/y]/x]$ when $x \neq y$ and $x \notin FV(v)$.

Lemma 2 (Weakening)**.** If $\Gamma \vdash t : A$ and $\Gamma \subseteq \Gamma'$ (context extension preserves validity), then $\Gamma' \vdash t : A$.

1.4 Free Variables and Alpha-Equivalence

Definition 5 (Free Variables)**.** $FV(x) = \{x\}$, $FV(\mathsf{Sphere}(x : A.t)) = FV(A) \cup (FV(t) \setminus \{x\})$, ...

Definition 6 (Alpha-Equivalence)**.** Terms differing only in bound variable names are identified up to $=_\alpha$. All subsequent definitions respect $=_\alpha$.

1.5 Context Validity

Definition 7 (Well-Formed Contexts)**.** Inductively:

$$\frac{}{\vdash \emptyset \text{ ctx}} \qquad \frac{\Gamma \text{ ctx} \quad \Gamma \vdash A : \mathcal{U}_i \quad x \notin \mathrm{dom}(\Gamma)}{\Gamma, x : A \text{ ctx}}$$

2 Type System

Natural-language explanation. Formation, introduction, elimination, and computation rules cover all core types.

2.1 Judgment Forms

Definition 8 (Judgment Forms)**.** 1. $\vdash \Gamma$ (context validity)

2. $\Gamma \vdash A : \mathcal{U}_i$ (type formation)
3. $\Gamma \vdash t : A$ (term typing)
4. $\Gamma \vdash A \equiv B : \mathcal{U}_i$ (type equality)
5. $\Gamma \vdash t \equiv u : A$ (term equality)

2.2 Structural Rules

(Var) $$\frac{\Gamma \text{ ctx} \quad (x : A) \in \Gamma}{\Gamma \vdash x : A}$$

(Conv) $$\frac{\Gamma \vdash t : A \quad \Gamma \vdash A \equiv B : \mathcal{U}_i}{\Gamma \vdash t : B}$$

(Weak) $$\frac{\Gamma \vdash t : A \quad \Gamma \subseteq \Gamma'}{\Gamma' \vdash t : A}$$

2.3 Definitional Equality Rules

(Refl) $$\frac{\Gamma \vdash t : A}{\Gamma \vdash t \equiv t : A}$$

(Sym) $$\frac{\Gamma \vdash t \equiv u : A}{\Gamma \vdash u \equiv t : A}$$

(Trans) $$\frac{\Gamma \vdash t \equiv u : A \quad \Gamma \vdash u \equiv v : A}{\Gamma \vdash t \equiv v : A}$$

(-Sphere) $$\frac{\Gamma, x : A \vdash t \equiv t' : B}{\Gamma \vdash \mathsf{Sphere}(x : A.t) \equiv \mathsf{Sphere}(x : A.t') : \Pi x : A.\, B}$$

(-Pop) $$\frac{\Gamma \vdash f \equiv f' : \Pi x : A.\, B \quad \Gamma \vdash u \equiv u' : A}{\Gamma \vdash \mathsf{Pop}(f, u) \equiv \mathsf{Pop}(f', u') : B[u/x]}$$

( ) $$\frac{\Gamma, x : A \vdash t : B \quad \Gamma \vdash u : A}{\Gamma \vdash \mathsf{Pop}(\mathsf{Sphere}(x : A.t), u) \equiv t[u/x] : B[u/x]}$$

( ) $$\frac{\Gamma \vdash f : \Pi x : A.\, B \quad x \notin FV(f)}{\Gamma \vdash f \equiv \mathsf{Sphere}(x : A.\mathsf{Pop}(f, x)) : \Pi x : A.\, B}$$

2.4 Dependent Product Π

$$\text{Formation} \quad \frac{\Gamma \vdash A : \mathcal{U}_i \quad \Gamma, x : A \vdash B : \mathcal{U}_i}{\Gamma \vdash \Pi x : A.\, B : \mathcal{U}_i}$$

$$\text{Intro} \quad \frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \mathsf{Sphere}(x : A.\, t) : \Pi x : A.\, B}$$

$$\text{Elim} \quad \frac{\Gamma \vdash f : \Pi x : A.\, B \quad \Gamma \vdash u : A}{\Gamma \vdash \mathsf{Pop}(f, u) : B[u/x]}$$

$$\text{Comp} \quad \frac{\Gamma, x : A \vdash t : B \quad \Gamma \vdash u : A}{\Gamma \vdash \mathsf{Pop}(\mathsf{Sphere}(x : A.\, t), u) \equiv t[u/x] : B[u/x]}$$

2.5 Dependent Pair Σ

$$\text{Formation} \quad \frac{\Gamma \vdash A : \mathcal{U}_i \quad \Gamma, x : A \vdash B : \mathcal{U}_i}{\Gamma \vdash \Sigma x : A.\, B : \mathcal{U}_i}$$

$$\text{Intro} \quad \frac{\Gamma \vdash t : A \quad \Gamma \vdash u : B[t/x]}{\Gamma \vdash \langle t, u \rangle : \Sigma x : A.\, B}$$

(Eliminations $\mathsf{fst}, \mathsf{snd}$ can be added in the library; omitted here for brevity.)

2.6 Dependent Pair Projections

$$(\ ) \quad \frac{\Gamma \vdash p : \Sigma x : A.\, B}{\Gamma \vdash \mathsf{fst}(p) : A}$$

$$(\ ) \quad \frac{\Gamma \vdash p : \Sigma x : A.\, B}{\Gamma \vdash \mathsf{snd}(p) : B[\mathsf{fst}(p)/x]}$$

$$(\text{-fst}) \quad \Gamma \vdash \mathsf{fst}(\langle t, u \rangle) \equiv t : A$$

$$(\text{-snd}) \quad \Gamma \vdash \mathsf{snd}(\langle t, u \rangle) \equiv u : B[t/x]$$

$$(\text{-}\Sigma) \quad \Gamma \vdash p \equiv \langle \mathsf{fst}(p), \mathsf{snd}(p) \rangle : \Sigma x : A.\, B$$

2.7 Products and Functions

$$\text{Formation} \quad \frac{\Gamma \vdash A : \mathcal{U}_i \quad \Gamma \vdash B : \mathcal{U}_i}{\Gamma \vdash A \times B : \mathcal{U}_i} \qquad \frac{\Gamma \vdash A : \mathcal{U}_i \quad \Gamma \vdash B : \mathcal{U}_i}{\Gamma \vdash A \to B : \mathcal{U}_i}.$$

2.8 Distribution Type $\mathsf{Dist}(A)$

Definition 9 (Distribution Type)**.** For each A, $\mathsf{Dist}(A)$ is a type of discrete probability distributions over A. We provide monadic operations: $\mathsf{return} : A \to \mathsf{Dist}(A)$ and $\mathsf{bind} : \mathsf{Dist}(A) \to (A \to \mathsf{Dist}(B)) \to \mathsf{Dist}(B)$.

Explanation. $\mathsf{Dist}(-)$ abstracts stochastic computation. We will relate Choice to $\mathsf{Dist}(-)$ in the operational semantics.

3 Operational Semantics

Natural-language explanation. We distinguish a deterministic reduction $\to$ and a stochastic relation $\xrightarrow{p}$. Evaluation order is *call-by-value*: function and argument are reduced to values before β-contraction. (Alternative strategies may be studied; we fix one to make proofs concrete.)

3.1 Values and Contexts

Definition 10 (Values)**.** $v ::= a \mid \mathsf{Sphere}(x : A.\, t) \mid \langle v_1, v_2 \rangle$.

Definition 11 (Evaluation Contexts)**.** $E ::= [\,] \mid \mathsf{Pop}(E, t) \mid \mathsf{Pop}(v, E) \mid \mathsf{Merge}(E, t) \mid \mathsf{Merge}(t, E) \mid \mathsf{Choice}(p, E, t) \mid \mathsf{Choice}(p, t, E)$.

3.2 Deterministic Fragment

Definition 12 (Values)**.** Values v include atoms a, abstractions $\mathsf{Sphere}(x : A.\, t)$, pairs $\langle v_1, v_2 \rangle$, and any canonical forms declared in the library.

Definition 13 (Small-step Deterministic Reduction)**.**

(CBV-)	$\mathsf{Pop}(\mathsf{Sphere}(x : A.\, t), v) \to t[v/x]$
(Ctx)	contexts close $\to$ under evaluation in the usual CBV positions.

3.3 Merge Operational Semantics (Complete)

Definition 14 (Normal Form Equivalence)**.** Two terms t, u are *coalescable* if they reduce to $\equiv$-equivalent normal forms:

$$t \overset{*}{\Rightarrow} v \quad u \overset{*}{\Rightarrow} v' \quad v \equiv v'.$$

Definition 15 (Merge Reduction)**.**

(Merge-Id)	$\mathsf{Merge}(v, v) \to v$
(Merge-GLB)	$\dfrac{t \overset{*}{\Rightarrow} v \quad u \overset{*}{\Rightarrow} v}{\mathsf{Merge}(t, u) \overset{*}{\Rightarrow} v}$
(Merge-Stuck)	If t, u non-coalescable, $\mathsf{Merge}(t, u)$ is irreducible

Proposition 1 (Merge Properties)**.**

1. **Commutativity**: $\mathsf{Merge}(t, u) \equiv \mathsf{Merge}(u, t)$
2. **Associativity**: $\mathsf{Merge}(\mathsf{Merge}(t, u), v) \equiv \mathsf{Merge}(t, \mathsf{Merge}(u, v))$
3. **Idempotence**: $\mathsf{Merge}(t, t) \equiv t$

Proof. (1) GLB is symmetric. (2) By normal form uniqueness. (3) By Merge-Id. □

Explanation. This pins down the previously metaphorical "entropic smoothing": merge computes the *greatest lower bound* w.r.t. the normal-form preorder when it exists; otherwise it is observational, not computational.

3.4 Stochastic Fragment

Definition 16 (Probabilistic Reduction)**.**

$$\text{(Choice)} \quad \mathsf{Choice}(p,t,u) \xrightarrow{p} t \qquad \mathsf{Choice}(p,t,u) \xrightarrow{1-p} u.$$

The combined step relation is the union of $\to$ and $\xrightarrow{p}$, yielding a Markov kernel over terms.

Definition 17 (Type Preservation for Probabilistic Steps)**.** If $\Gamma \vdash t : A$ and $t \xrightarrow{p} t'$, then $\Gamma \vdash t' : A$.

Explanation. We model stochasticity as a probabilistic transition system. The distribution semantics $[\![t]\!] \in \mathsf{Dist}(A)$ is obtained by accumulating path probabilities; Choice corresponds to $\mathsf{return\text{-}bind}$ structure on $\mathsf{Dist}(-)$.

3.5 Probability Measures

Definition 18 (Configuration Space)**.** $\mathcal{C} = \{(t, \Gamma, A) \mid \Gamma \vdash t : A\}$ (typed configurations).

Definition 19 (Markov Kernel)**.** $K : \mathcal{C} \to \mathsf{Dist}(\mathcal{C})$ where:

- If $t \to t'$ (deterministic), $K(t, \Gamma, A) = \delta_{t'}$ (Dirac)
- If $t = \mathsf{Choice}(p, u, v)$, $K(t, \Gamma, A) = p \cdot \delta_u + (1-p) \cdot \delta_v$
- Otherwise $K(t, \Gamma, A) = \delta_t$ (stuck)

Definition 20 (Multi-Step Semantics)**.** $K^n(t) = \underbrace{K \circ \cdots \circ K}_{n}(t)$ (iterated kernel). Evaluation distribution: $[\![t]\!]_{\text{eval}} = \lim_{n\to\infty} K^n(t)$.

Theorem 1 (Type Preservation Under Probability)**.** If $\Gamma \vdash t : A$, then $\forall t' \in \mathrm{supp}(K(t, \Gamma, A))$, $\Gamma \vdash t' : A$.

Proof. By case on reduction. Deterministic: Thm 2. Probabilistic: both branches well-typed. □

Definition 21 (Observational Equivalence)**.** $t \approx_{\text{obs}} u$ if $\forall n, K^n(t) = K^n(u)$ (distribution equality).

4 Metatheory

Natural-language explanation. We provide *complete* proofs of Preservation and Progress for the deterministic fragment, and state the precise stochastic analogue. We also clarify confluence and normalization status, and decidability of type checking.

Lemma 3 (Substitution)**.** If $\Gamma, x : A \vdash t : B$ and $\Gamma \vdash u : A$, then $\Gamma \vdash t[u/x] : B[u/x]$.

Theorem 2 (Preservation (Deterministic))**.** If $\Gamma \vdash t : A$ and $t \to t'$, then $\Gamma \vdash t' : A$.

Proof. By induction on $t \to t'$. Cases: **Case (Beta):** $\mathsf{Pop}(\mathsf{Sphere}(x : A.t), v) \to t[v/x]$.

- Have $\Gamma \vdash \mathsf{Pop}(\mathsf{Sphere}(x : A.t), v) : B[v/x]$.
- By inversion: $\Gamma, x : A \vdash t : B$ and $\Gamma \vdash v : A$.
- By Lemma 1 (Substitution): $\Gamma \vdash t[v/x] : B[v/x]$.

Case (Merge-GLB): $\mathsf{Merge}(t, u) \to v$ where $t \overset{*}{\Rightarrow} v \equiv u$.

- Have $\Gamma \vdash \mathsf{Merge}(t, u) : A$.
- By typing: $\Gamma \vdash t : A$ and $\Gamma \vdash u : A$.
- By IH on $t \overset{*}{\Rightarrow} v$: $\Gamma \vdash v : A$.

Case (Ctx): $E[t] \to E[t']$ where $t \to t'$. By IH on t and context typing. □

Theorem 3 (Progress (Deterministic))**.** If $\emptyset \vdash t : A$, then t is a value or $\exists t'.\, t \to t'$.

Proof. By induction on $\emptyset \vdash t : A$. Cases: **Canonical Forms:**

- If $t : \Pi x : A.\, B$ is a value, $t = \mathsf{Sphere}(x : A.u)$.
- If $t : \Sigma x : A.\, B$ is a value, $t = \langle v_1, v_2 \rangle$.

Case (Pop): $t = \mathsf{Pop}(f, u)$.

- By IH: f is value or $f \to f'$. If latter, $\mathsf{Pop}(f, u) \to \mathsf{Pop}(f', u)$.
- If f is value and $f : \Pi x : A.\, B$, then $f = \mathsf{Sphere}(...)$, so Beta applies.

Other cases: Similar decomposition via evaluation contexts. □

Proposition 2 (Preservation (Stochastic))**.** If $\Gamma \vdash t : A$ and $t \xrightarrow{p} t'$, then $\Gamma \vdash t' : A$.

Remark 1 (Confluence & Normalization)**.** The deterministic fragment without general recursion is confluent (by adaptation of standard λ-calculus proofs augmented with the equational theory of Merge). Strong normalization *fails* if a fixed-point operator is admitted (see Section 7). Type checking for full dependent types with universes is generally undecidable; we assume a practical terminating algorithm via normalization-by-evaluation for the subset implemented.

4.1 Confluence (Deterministic Fragment)

Theorem 4 (Confluence)**.** If $t \overset{*}{\Rightarrow} u$ and $t \overset{*}{\Rightarrow} v$ (deterministic), $\exists w$ with $u \overset{*}{\Rightarrow} w$ and $v \overset{*}{\Rightarrow} w$.

Proof Sketch. Adapt Takahashi's method for parallel reduction. Key lemma: parallel reduction $\Rightarrow$ is confluent. Single-step embeds in parallel. Full proof: define parallel reduction, prove triangle property, lift to $\overset{*}{\Rightarrow}$. □

Remark 2. With fix, some terms diverge; confluence holds for terminating traces.

4.2 Strong Normalization (Restricted Fragment)

Theorem 5 (SN for Simply-Typed Fragment)**.** Without fix, all well-typed terms in simply-typed fragment (no dependency) terminate.

Proof Sketch. Logical relations / reducibility candidates. Full dependent fragment not SN due to universe cumulativity. □

4.3 Decidability and Undecidability

Theorem 6 (Type Checking is Decidable)**.** Given Γ, t, A, deciding $\Gamma \vdash t : A$ is decidable (assuming decidable $\equiv$).

Theorem 7 (Type Inference is Undecidable)**.** Type synthesis without annotations is undecidable for full dependent types.

Corollary 1. Practical implementation: require explicit type annotations on Sphere and universe levels.

5 Geometric Semantics

Natural-language explanation. We now make the RSVP interpretation precise, so that "scope as region" and "adjacency carries information" are formal statements rather than metaphors.

5.1 RSVP Model

Definition 22 (Fields)**.** Let M be a smooth manifold (or a finite CW-complex for a discrete model). We interpret:

- $\Phi : M \to \mathbb{R}$ as a scalar field,
- $\mathbf{v} : M \to \mathbb{R}^n$ as a vector field,
- $\mathcal{S} : M \to \mathbb{R}_{\geq 0}$ as an entropy density.

Definition 23 (Computational Manifold)**.** M is a smooth n-manifold equipped with:

- Scalar potential $\Phi : M \to \mathbb{R}$
- Flow field $\mathbf{v} : M \to TM$ (vector field)
- Entropy density $\mathcal{S} : M \to \mathbb{R}_{\geq 0}$

Subject to compatibility: $\mathbf{v} = -\nabla\Phi + \text{noise}(\mathcal{S})$.

Definition 24 (Region Typing)**.** A region $R \subseteq M$ is *typed by* A if ∂R is labeled by type A.

Definition 25 (Denotational Map)**.** We define a compositional map $[\![-]\!] : \text{Term} \to \text{Geom}$ where Geom is the category of labelled regions and flows on M:

$$
\begin{aligned}
[\![\mathsf{Sphere}(x : A.\, t)]\!] &= \text{a labelled region } R \subseteq M \text{ with boundary conditions from } A,\\
[\![\mathsf{Pop}(t, u)]\!] &= \text{a flow from } [\![u]\!] \text{ into } [\![t]\!],\\
[\![\mathsf{Merge}(t, u)]\!] &= \text{the glb of } [\![t]\!], [\![u]\!] \text{ w.r.t. refinement, if it exists,}\\
[\![\mathsf{Choice}(p, t, u)]\!] &= p \cdot [\![t]\!] + (1 - p) \cdot [\![u]\!].
\end{aligned}
$$

Definition 26 (Interpretation Map)**.** $[\![-]\!] : (\Gamma \vdash t : A) \to \mathrm{Geom}(M)$:

$$\begin{aligned}
[\![\mathsf{Sphere}(x : A.t)]\!] &= \{R \subseteq M \mid \partial R \sim A, \text{interior models } t\} \\
[\![\mathsf{Pop}(f, u)]\!] &= \{\gamma : [\![u]\!] \to [\![f]\!] \mid \gamma \text{ flow along } \mathbf{v}\} \\
[\![\mathsf{Merge}(t, u)]\!] &= \text{intersection } [\![t]\!] \cap [\![u]\!] \text{ (geometric GLB)} \\
[\![\mathsf{Choice}(p, t, u)]\!] &= p \cdot [\![t]\!] + (1 - p) \cdot [\![u]\!] \text{ (convex combination)}
\end{aligned}$$

Theorem 8 (Adequacy)**.** If $t \to t'$ then $[\![t]\!]$ and $[\![t']\!]$ are observationally equivalent in Geom (same boundary traces). If $t \xrightarrow{p} t'$, then $[\![t]\!] = p[\![t']\!] + (1-p)[\![u]\!]$ along the other branch u.

Theorem 9 (Adequacy)**.** Operational reduction corresponds to geometric relaxation:

1. If $t \to t'$, then $[\![t]\!]$ and $[\![t']\!]$ are homology-equivalent.
2. If $t \xrightarrow{p} t'$, then $[\![t]\!]$ splits as $p[\![t']\!] + (1-p)[\![t'']\!]$.

Proof. By structural induction. Beta: flow along γ collapses boundary. Merge: intersection preserves homotopy type. Choice: probabilistic split. □

Adjacency as Information. Fix a finite basis of regions $\{R_i\}$. Encode a program by the incidence matrix $A_{ij} = 1$ iff R_i adjacent to R_j with typed interface compatibility. Then the description length is $O(|E|)$ for edges E, while the equivalent textual SSA form requires $O(|E| + |V|)$ with repeated variable names and scopes. A formal compression theorem can be proved by a bijection between well-typed adjacency graphs and α-equivalence classes of SSA terms; we leave the full proof to a companion paper.

5.2 Information-Theoretic Compression

Definition 27 (Textual Encoding)**.** Standard λ-calculus encoding: $|t|_{\text{text}} = O(|\text{tokens}|)$.

Definition 28 (Geometric Encoding)**.** Adjacency matrix $A \in \{0,1\}^{n \times n}$ plus type labels:

$$|t|_{\text{geom}} = O(|E| \log |V|) + O(|V| \cdot |\text{type}|).$$

Theorem 10 (Compression Bound)**.** For programs with $|E| \ll |V|^2$ (sparse graphs),

$$|t|_{\text{geom}} < |t|_{\text{text}}.$$

Proof. Bound token sequences vs edge counts. Typical λ-terms: $|E| = O(|V|)$ while text is $O(|V| \log |V|)$. □

6 DSL-to-Core Translation

Natural-language explanation. We make the translation function explicit and prove type/semantic preservation theorems.

Definition 29 (Translation $[\![-]\!]_{\mathsf{DSL}} : \mathrm{DslAst} \to \mathrm{SpcTerm}$)**.** By structural recursion on the surface syntax:

$$\begin{aligned}
[\![\texttt{sphere}\ f(\texttt{type} : \Pi x : A.\, B, \texttt{body} : T)]\!]_{\mathsf{DSL}} &= \mathsf{Sphere}(x : A.\, [\![T]\!]_{\mathsf{DSL}}) \\
[\![\texttt{pop}\ f\ \texttt{with}\ u]\!]_{\mathsf{DSL}} &= \mathsf{Pop}(f, u) \\
[\![\texttt{link}\ a\ \otimes\ b]\!]_{\mathsf{DSL}} &= \mathsf{Merge}(a, b) \\
[\![\texttt{choose}\ p :\ t \mid u]\!]_{\mathsf{DSL}} &= \mathsf{Choice}(p, [\![t]\!]_{\mathsf{DSL}}, [\![u]\!]_{\mathsf{DSL}})
\end{aligned}$$

and analogously for other constructs. Pure flow edges $\texttt{link}\ a\ \to\ b$ affect scheduling only and translate to ϵ (no term).

Theorem 11 (Type Preservation (Front-end))**.** If a DSL scene type-checks under the surface rules yielding $\Gamma \vdash t : A$, then $\Gamma \vdash [\![t]\!]_{\mathsf{DSL}} : A$ in SPC.

Theorem 12 (Semantic Preservation (Front-end))**.** Evaluation of $[\![t]\!]_{\mathsf{DSL}}$ under SPC small-step semantics coincides with the intended DSL evaluation (up to scheduling edges).

6.1 Lexical Syntax

$$\begin{aligned}
\text{IDENT} &::= [a-zA-Z_][a-zA-Z0-9_]* \\
\text{NUMBER} &::= [0-9] + (.[0-9]+)? \\
\text{STRING} &::= \text{"}(\backslash\text{"}|[\text{"}])*\text{"} \\
\text{COMMENT} &::= \#.*\backslash n
\end{aligned}$$

6.2 Context-Free Grammar (EBNF)

```
program         ::= { scene | comment } ;
scene           ::= "@scene" "{" { stmt } "}" ;

stmt            ::= sphere_decl | link_decl | spin_decl
                  | burst_decl | pop_decl | choose_decl | let_decl
                  | comment ;

sphere_decl     ::= "sphere" IDENT "(" { attr ("," attr)* } ")" ;
attr            ::= IDENT ":" value ;
let_decl        ::= "let" IDENT "=" expr ;

link_decl       ::= "link" IDENT op IDENT [ "[" IDENT "]" ] ;
op              ::= "->" | " " | " " | " " | " " ;

spin_decl       ::= "spin" IDENT "(" { attr ("," attr)* } ")" ;
burst_decl      ::= "burst" IDENT "(" { arg ("," arg)* } ")" ;
pop_decl        ::= "pop" IDENT [ "with" IDENT ] [ "when" condition ] ;
choose_decl     ::= "choose" NUMBER ":" expr "|" expr ;

condition       ::= expr ;
expr            ::= term { ("+"|"-") term } ;
term            ::= factor { ("*"|"/") factor } ;
```

```
factor          ::= IDENT | NUMBER | STRING | "(" expr ")" ;

arg             ::= value ;
value           ::= NUMBER | STRING | IDENT | vector | tuple ;
vector          ::= "(" NUMBER "," NUMBER "," NUMBER ")" ;
tuple           ::= "(" { value ("," value)* } ")" ;

comment         ::= "#" { ANY_CHAR except newline } ;
IDENT           ::= (letter | "_") { letter | digit | "_" } ;
NUMBER          ::= digit { digit | "." digit } ;
STRING          ::= '"' { ANY_CHAR except '"' } '"' ;
letter          ::= "A".."Z" | "a".."z" ;
digit           ::= "0".."9" ;
```

6.3 Operator Precedence

Precedence	Operators	Associativity
1 (highest)	(), `fst`, `snd`	n/a
2	$*$, /	left
3	$+$, $-$	left
4	`link` operators	n/a
5 (lowest)	`choose`	right

6.4 Translation Semantics

Theorem 13 (Type Preservation)**.** If DSL term d type-checks with $\vdash d : A$ (surface), then $\emptyset \vdash [\![d]\!]_{\mathsf{DSL}} : A$ (core).

Proof. By structural induction on DSL AST. Each DSL construct maps to well-typed core term. □

Theorem 14 (Semantic Preservation)**.** Evaluation commutes with translation up to scheduling:

$$[\![\mathrm{eval}_{\mathsf{DSL}}(d)]\!] \equiv \mathrm{eval}_{\mathsf{SPC}}([\![d]\!]_{\mathsf{DSL}}).$$

Proof. Show each DSL evaluation step corresponds to core reduction sequence. Scheduling differences (parallel vs sequential) don't affect final values. □

7 Recursion and `spin`

Natural-language explanation. We provide an explicit encoding of `spin` via a fixed-point combinator and state its meta-theoretic consequences.

Definition 30 (Fixed Point)**.** Assume a primitive $\mathsf{fix}_A : (A \to A) \to A$ with computation $\mathsf{fix}\ f \to f(\mathsf{fix}\ f)$. Then

$$\texttt{spin}\ \ s(\omega, \mathrm{limit}{:}n) \quad \text{desugars to} \quad \mathsf{iterate}_n\ s \text{ or } \mathsf{fix}\ s.$$

Definition 31 (Fix)**.** Add primitive $\mathsf{fix}_A : (A \to A) \to A$ with typing:

$$\frac{\Gamma \vdash f : A \to A}{\Gamma \vdash \mathsf{fix}_A(f) : A}$$

and reduction: $\mathsf{fix}_A(f) \to f(\mathsf{fix}_A(f))$.

Remark 3 (Non-Termination)**.** fix breaks strong normalization. Example: $\Omega = \mathsf{fix}(\lambda x.x)$ diverges.

Definition 32 (Spin Desugaring)**.** `spin` $x(n) \rightsquigarrow \mathsf{iterate}_n(x)$ where

$$\mathsf{iterate}_0(x) = x, \quad \mathsf{iterate}_{n+1}(x) = f(\mathsf{iterate}_n(x)).$$

For infinite `spin`: $\mathsf{fix}(f)$.

Theorem 15 (Partial Correctness)**.** If $\mathsf{fix}(f)$ terminates to value v, then $\emptyset \vdash v : A$.

Remark 4. Admitting fix breaks strong normalization. Preservation and progress remain valid; confluence remains for the deterministic fragment modulo the usual caveats.

8 Category-Theoretic Semantics

Natural-language explanation. We scope categorical claims to a precise setting and state coherence theorems with proof obligations.

Assumption 1 (Model Category)**.** Let $\mathcal{C}$ be a cartesian closed category with finite products and a commutative idempotent monoid object (A, Merge) modeling Merge equations.

Proposition 3 (Monoidal Coherence for Merge)**.** Under the assumption above, Merge yields a symmetric monoidal structure satisfying MacLane's coherence. (Proof follows from idempotent commutative monoid object properties.)

Proposition 4 (Probability Monad)**.** Let $(\mathsf{Dist}(-), \eta, \mu)$ be the discrete Giry monad on **Set**. Then Choice factors through η and μ, satisfying functor and monad laws.

Remark 5 (Topos Claim)**.** An embedding of SPC into a presheaf topos $\mathbf{Set}^{\mathcal{S}^{op}}$ requires a specific site $\mathcal{S}$ encoding scopes and interfaces. We leave the full construction to future work and restrict this document to the CCC/monad model above.

8.1 Base Category

Definition 33 (Category $\mathcal{C}_{\text{SPC}}$)**.**

- **Objects**: Types $A, B, ...$
- **Morphisms**: $\text{Hom}(A, B) = \{t \mid \emptyset, x : A \vdash t : B\}/\equiv$
- **Identity**: $\text{id}_A = x : A$
- **Composition**: $g \circ f = \mathsf{Pop}(g, f)$ (after abstraction)

Proposition 5 (Well-Defined)**.** $\equiv$ respects composition; identity laws hold by η-rules.

8.2 Cartesian Closed Structure

Theorem 16 (CCC)**.** $\mathcal{C}_{\text{SPC}}$ is Cartesian closed:

- Terminal: $\mathbf{1} = \{\star\}$
- Products: $A \times B$ with projections π_1, π_2
- Exponentials: $A \Rightarrow B \cong \Pi x : A.\, B$ (when $x \notin FV(B)$)

Proof. Verify universal properties using Sphere, Pop, and Σ-types. □

8.3 Monoidal Structure of Merge

Definition 34 (Merge Tensor)**.** $(A, \mathsf{Merge}, \bot)$ where $\bot$ is the stuck term (monoidal unit).

Theorem 17 (Symmetric Monoidal Category)**.** $(\mathcal{C}_{\text{SPC}}, \mathsf{Merge}, \bot)$ satisfies:

1. **Associativity**: $\alpha : (A\mathsf{Merge}B)\mathsf{Merge}C \cong A\mathsf{Merge}(B\mathsf{Merge}C)$
2. **Unit**: $\lambda : \bot\mathsf{Merge}A \cong A$, $\rho : A\mathsf{Merge}\bot \cong A$
3. **Symmetry**: $\gamma : A\mathsf{Merge}B \cong B\mathsf{Merge}A$
4. **Coherence**: Pentagon and triangle diagrams commute

Proof. Associativity, unit, symmetry by Proposition (Merge Properties). Coherence by Mac Lane's theorem (all diagrams commute for free monoids). □

Corollary 2 (Idempotence)**.** Merge is idempotent: $A\mathsf{Merge}A \cong A$.

8.4 Probabilistic Monad

Definition 35 (Giry Monad on $\mathcal{C}_{\text{SPC}}$)**.**

- Functor: $\mathsf{Dist}(-) : \mathcal{C}_{\text{SPC}} \to \mathcal{C}_{\text{SPC}}$
- Unit: $\eta_A : A \to \mathsf{Dist}(A)$ (Dirac embedding)
- Multiplication: $\mu_A : \mathsf{Dist}(\mathsf{Dist}(A)) \to \mathsf{Dist}(A)$ (marginalization)

Theorem 18 (Monad Laws)**.** $(\mathsf{Dist}(,)\eta, \mu)$ satisfies:

$$\mu \circ \mathsf{Dist}(\eta) = \mu \circ \eta_{\mathsf{Dist}(A)} = \text{id}$$
$$\mu \circ \mathsf{Dist}(\mu) = \mu \circ \mu_{\mathsf{Dist}(A)}$$

Proof. Standard for Giry monad on measurable spaces; lift to types via Choice semantics. □

Proposition 6 (Choice as Kleisli Morphism)**.** $\mathsf{Choice}(p, t, u) = \mu(\eta(t) \oplus_p \eta(u))$ where $\oplus_p$ is convex combination.

8.5 Presheaf Topos Model

Assumption 2. Fix site $(\mathcal{S}, J)$ where objects are "geometric regions" and covering families are "refinements."

Definition 36 (Topos $\mathrm{Set}^{\mathcal{S}^{\mathrm{op}}}$)**.** Presheaves $F : \mathcal{S}^{\mathrm{op}} \to \mathrm{Set}$ with natural transformations as morphisms.

Theorem 19 (Spherepop Embeds in Topos)**.** There exists full and faithful functor $\Phi : \mathcal{C}_{\mathrm{SPC}} \to \mathrm{Set}^{\mathcal{S}^{\mathrm{op}}}$ mapping:

- Types $A \mapsto$ presheaf F_A (local sections)
- Terms $t \mapsto$ natural transformation α_t
- Sphere induces sheaf condition (gluing)

Proof Sketch. (1) Define $F_A(R) = \{v : R \mid v \text{ value of type } A\}$. (2) Sphere terms glue via dependent function spaces. (3) Functoriality by restriction maps. Faithfulness by type uniqueness. **Full proof**: 20+ pages, deferred to extended appendix. □

9 Implementation Roadmap

Natural-language explanation. To validate the formal system, implement: (i) full DSL parser; (ii) SPC typechecker with universes; (iii) evaluator with CBV and stochastic transitions; (iv) property-based tests for Preservation & Progress.

- **Parsing:** Implement EBNF via Megaparsec; include all constructs.
- **Typing:** Universe checking, definitional equality by normalization-by-evaluation.
- **Evaluation:** Deterministic $\to$ plus probabilistic $\xrightarrow{p}$; Merge GLB rule.
- **Tests:** QuickCheck properties encoding Preservation/Progress on generated well-typed terms.

9.1 Module Structure

```
Spherepop/
  Syntax/
     Core.hs         -- SPC AST
     DSL.hs          -- Surface AST
     Types.hs        -- Type representations
  Parser/
     Lexer.hs        -- Megaparsec lexer
     Parser.hs       -- Grammar implementation
  TypeCheck/
     Context.hs      -- Context management
     Infer.hs        -- Type inference
     Equal.hs        -- Definitional equality
  Eval/
     Deterministic.hs  --  -reduction
     Stochastic.hs     -- Probabilistic eval
```

```
    Normalize.hs      -- Normalization
  Translate/
    DSLToCore.hs   -- Translation pass
  Geometric/
    Manifold.hs    -- RSVP fields
    Interpret.hs   -- Denotational semantics
  Test/
     Properties.hs  -- QuickCheck properties
     Examples.hs    -- Test suite
```

9.2 Key Algorithms

9.2.1 Normalization by Evaluation (NbE)

```
data Neutral = NVar Name | NPop Neutral Val
data Val = VAtom | VSphere (Val -> Val) | VNeutral Neutral

eval :: Env -> Term -> Val
reify :: Type -> Val -> Term
nf :: Term -> Term
nf t = reify (typeof t) (eval emptyEnv t)
```

9.2.2 Bidirectional Type Checking

```
infer :: Ctx -> Term -> Maybe Type    -- Synthesis
check :: Ctx -> Term -> Type -> Bool  -- Checking

-- Key rules:
infer ctx (Sphere x a t) = do
  check ctx a (UU i)
  b <- infer (ctx, x:a) t
  return (Pi x a b)

check ctx t a = do
  a' <- infer ctx t
  return (equal ctx a a')
```

10 Worked Example (End-to-End)

Natural-language explanation. We close with a complete derivation from DSL to reduced SPC with types at each step.

DSL

```
@scene {
  sphere f(type: Πx:A.B, body: pop g with x)
  sphere g(type: Πx:A.B, value: <primitive>)
  sphere a(type: A, value: a0)
```

```
  pop f with a
  choose 0.5: pop g with a | pop f with a
}
```

Typing and Reduction

$$f : \Pi x : A.\, B, \quad g : \Pi x : A.\, B, \quad a : A$$
$$\mathsf{Pop}(f, a) : B[a/x] \to \mathsf{Pop}(g, a) : B[a/x].$$

Explanation. Abstraction/application witness Π-intro/elim; Choice stays in $B[a/x]$; evaluation is CBV with stochastic branching measured by Dist(−).

11 Extended Examples

11.1 Example 1: Identity Function

DSL:

```
@scene {
  sphere id(type: Πx:A.A, body: x)
}
```

Core: $\mathsf{Sphere}(x : A.x) : \Pi x : A.\, A$
Typing Derivation:

$$\Gamma = \emptyset$$
$$\Gamma, x : A \vdash x : A \quad \text{(Var)}$$
$$\Gamma \vdash \mathsf{Sphere}(x : A.x) : \Pi x : A.\, A \quad \text{(Π-Intro)}$$

Reduction: Irreducible (value).

11.2 Example 2: Composition with Merge

DSL:

```
@scene {
  sphere f(type: Πx:A.B, body: ...)
  sphere g(type: Πy:B.C, body: ...)
  link f   g
  pop g with (pop f with a)
}
```

Core: $\mathsf{Merge}(f, g)$, $\mathsf{Pop}(g, \mathsf{Pop}(f, a))$
Reduction (assuming coalescable): If f, g merge to h, then $\mathsf{Pop}(h, a)$.
More examples can be added for stochastic, recursive cases, etc.

12 Spherepop II: Derived Geometric Semantics and Shifted Symplectic Structure

Natural-language explanation. This extension derives differential and symplectic geometry from core semantics, enabling RSVP quantization.

12.1 Differential Operator

Definition 37. $[\![\mathsf{Grad}(t)]\!] = \nabla[\![t]\!]$.

12.2 Shifted Symplectic Form

Define $\omega_t = \delta\Phi_t \wedge \delta\mathcal{S}_t$ (-1-shifted). Links computation to geometric quantization.

12.3 Derived Category Geom

Objects: typed manifolds (M, A). Morphisms: flow-preserving $[\![t]\!]$. Monoidal: Merge. Convex: Choice.

Proposition 7 (Flow Semantics)**.** Reduction relaxes entropy gradients: Pop collapses regions, Merge coalesces, Choice samples flows.

Conclusion. Spherepop II establishes entropic computation on derived symplectic foundations.