DOLICE

Causal Horizon Funeral Procession

葬　列

Anchored in the artwork “Causal Sets / 葬列,” reframing time not as a flow but as a partial order of causal relations — Complete Edition v10. The five axioms of the projection rule f are made computable, and the central thesis — we are neither born nor perishing, and inside the black-hole horizon we form a funeral procession — is restated as facts of order theory.

COMPLETE EDITION · v10 · EN

Masaki Hirokawa（廣川政樹）

dolice.net

Contents

	Artwork Caption	3
	Abstract	6
0	The Central Thesis	7
1	Projection Rule f (I/O Specification)	8
2	Horizon, Controller, and Counterexamples	12
3	Protocol Mapping (Production → Submission → Publication)	16
4	v9 — A Minimal Axiom System for the Projection Rule f	17
5	v10 — Inside the Horizon	22
6	Strings and Branes — A Reading from Superstring and M-theory	26
7	Mandala and the Tree of Life — Concentric Projection of an Order	28
8	Unborn, Undying — Nirvāṇa Restated in Order Theory	30
9	Cultural Evolution and Diversity — A4 in Time	37
10	Testability and Limits	39
11	Closing — What v10 Closes, and What It Opens	40
12	Plate Gallery (v1–v10)	41
	Appendix A — Running the Code	53
	Appendix B — Glossary	54
	References	55

Artwork Caption



Causal Sets / 葬列 (2023) — three veiled figures joined through a single red rose: each isolated within its cell, yet bound into one procession by inescapable threads of fate.

作品名 / Title	Causal Sets / 葬列
制作年 / Year	2023 年 4 月 / Apr 2023
サイズ / Dimensions	41 × 61 cm（縦 × 横 / H × W）
メディウム / Medium	デジタルメディア / Digital media
技法 / Technique	フォトコラージュ / Photo collage
作品ページ / Artwork page	https://dolice.net/art/causal-sets/

“Causal Sets” draws inspiration from Eastern mysticism and the theory of causal sets — a conceptual bridge between quantum mechanics and general relativity. The artwork explores the simultaneity of cause and effect, illustrating how events of the past, present, and future unfold in an instant, thereby challenging the traditional notion of temporal continuity. Each being is depicted as isolated within its own cell, yet interconnected through inevitable bonds of fate. These invisible threads of causality shape our collective existence, revealing that even in perceived solitude, we are never truly separate from the whole.

The piece conveys a profound message: every individual is unique and indispensable. Each presence contributes irrevocably to the vast transformations of the world, serving as a vital tessera in the ever-shifting mosaic of the cosmos. Through the metaphor of a funeral procession stretching endlessly through a sea of stars, the work invites reflection on the ephemeral nature of life. It urges us to treasure each fleeting moment and every meaningful encounter — each karmic thread of connection (en) — in this transient world.

— Related Work (Supporting Reference)

The repetition and nesting of "Archetypus" provide a supporting frame for rank-nesting (v5) and for the mandala projection of §7, where the tiers of the order appear as concentric rings.



Archetypus / 元型 (2020) — a nested image of repeated bird-beak masks; an endless, mirror-like recurrence.

作品名 / Title	Archetypus / 元型
制作年 / Year	2020 年 7 月 / Jul 2020
サイズ / Dimensions	123 × 99 cm（縦 × 横 / H × W）
メディウム / Medium	デジタルメディア / Digital media
技法 / Technique	フォトコラージュ / Photo collage
作品ページ / Artwork page	https://dolice.net/art/archetypus/

This artwork, titled “Archetypus,” explores the theme of “A Return to the Archetypal and Repetition,” featuring repeated reproductions of a woman wearing a bird mask — an expression of humanity's fierce desires. The logic that sustains madness is indispensable to art, and indeed to human civilization; yet for the artist, logic is no more than a means of restraining the senses. The coexistence of the two thus gives rise to a repetition like that of facing mirrors — endlessly looping within nested structures. There is no end to it; only a nested world looping on. Perhaps human history, culture, wars and conflicts, and even the cosmos itself, have been constructed in just this way.

Abstract

This paper reframes the artwork “Funeral Procession” as a structure fixed near the horizon B of a black-hole spacetime, and treats time not as a flow but as a partial order of causal relations. The projection rule f, built up empirically across v1–v9, was reconstructed at v9 from five minimal axioms (A1–A5). This Complete Edition v10 makes that axiom system computable — a sprinkled causal set, a horizon modelled as a null surface, and four parameters θ = (p, q, σ, κ) — obtains the stable regime Ω_stable as a phase diagram, and derives the A5 bound κ_max numerically from maximal flow (bit threads). It then restates the central thesis — we are neither born nor perishing; we stand inside the black-hole horizon; and every one of us forms a funeral procession — as three facts of order theory: the window-edge theorem (first and last elements are artefacts of the observation window), the vanishing proper time τ = 0 on the horizon, and Dilworth's chain decomposition. Correspondences with superstring and M-theory (T-duality, a world on a brane), with the mandala and the Tree of Life (concentric projection of an order), and with nirvāṇa (unborn, unending) are stated with the line between theorem and metaphor drawn explicitly. Every line of code is annotated, and every figure is regenerated from that code.

KEYWORDS causal set theory · event horizon · holographic principle · bit threads · Dilworth's theorem · minimal axiom system · phase diagram · T-duality · mandala · Tree of Life · unborn-unending · cultural evolution

SECTION 0

The Central Thesis

What this paper finally wants to say fits in three sentences. For each thesis we separate the order-theoretic fact, the physical correspondence, and the correspondence in Eastern thought, so that it is explicit at which layer what is shown. The aim is not to be believed, but to measure how far each thesis can be computed.

> P 1
>
> We are neither born nor perishing.

Order-theoretic fact: inside a finite observation window there are always elements without a past (beginnings) and without a future (endings), but if the window is held fixed while the causal set is enlarged, their fraction tends to zero (§8.1, Fig. 8). Beginning and end are properties of the window's edge, not of the order. Physics: no proper time passes on the horizon ($\tau = 0$, §8.2). Eastern thought: the Heart Sūtra's "neither born nor extinguished."

> P 2
>
> This world lies inside the event horizon of a black hole.

Order-theoretic fact: splitting the causal set by the null surface u = u_h, threads can cross only from outside to inside (§2, §5.1). Seen from within, the horizon is an uncrossable past; seen from without, a boundary that can only be read. Physics: the holographic principle — the bulk is reconstructed from boundary data (§5.2). The horizon treated here lies in the interior of Ω_stable, where A1–A5 hold at once — an idealisation of a static Schwarzschild horizon. Eastern thought: the phenomenal world unfolds from the storehouse consciousness (§4.6).

> P 3
>
> Every one of us forms a funeral procession.

Order-theoretic fact: Dilworth's theorem. Any finite partial order can be partitioned into as many chains (totally ordered columns) as its width, and every element belongs to exactly one chain (§8.4, Fig. 10). No one stands outside the column. Physics: bit threads — the maximal number of independent threads across the horizon equals the minimum cut (§5.2). Eastern thought: dependent origination. An element is a bundle of relations and has no substance outside them (non-self).

SECTION 1

Projection Rule f (I/O Specification)

f is the minimal rule mapping the Model/Controller state into View outputs. “Minimal” does not mean restricting expression; it means traceable freedom: the same input yields the same output, and output differences remain attributable to specific factors.

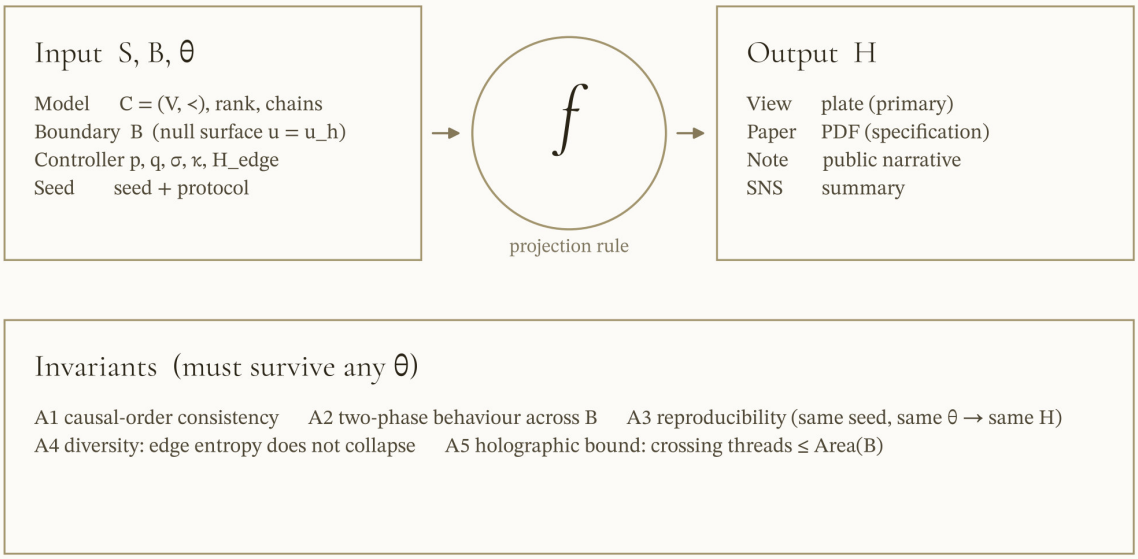

Fig. 1 I/O of the projection rule f and its invariants.

1.1 Internal State (Minimal Definition)

```
Input model : S = (V, ≺), rank, chains
Boundary    : B  (null surface u = u_h)
Controller  : θ = (p, q, σ, κ), H_edge
Seed        : seed + protocol
Output      : H = f(S; B, θ)
```

In v10 the Model is made concrete as a causal set sprinkled uniformly in light-cone coordinates. Sprinkling is the standard procedure of causal set theory: a Poisson process drops a number of points proportional to the spacetime volume, and the order $\prec$ is fixed by the light cones. No other structure — distance, clock, coordinate — is attached to an element. We adopt the principle of causal set theory as it stands: order plus number equals geometry.

COMMON SETUP

```
# Numerical foundation: point clouds, distances and randomness all go through this library
import numpy as np
# Used for every plate in this paper
import matplotlib.pyplot as plt
# Layout helper for arranging several panels on one plate
from matplotlib.gridspec import GridSpec
# Graph library used for max-flow / min-cut (bit threads)
import networkx as nx
# Natural log and pi (needed for entropy and angles)
import math
# Decorator to declare the Controller θ as a named bundle of values
from dataclasses import dataclass
```

SPRINKLING A CAUSAL SET

```
def sprinkle_diamond(n=320, seed=7):
    # Reproducibility: all randomness comes from a seed (premise of A3)
    rng = np.random.default_rng(seed)
    # Scatter light-cone coordinates u, v uniformly on [0, 1) (approximate Poisson sprinkling)
    u = rng.random(n)
    # Same for v; independent u, v fall uniformly on the diamond
    v = rng.random(n)
    # Time coordinate t = (u + v) / √2; larger means "future"
    t = (u + v) / math.sqrt(2.0)
    # Space coordinate x = (v − u) / √2; the lateral extent
    x = (v - u) / math.sqrt(2.0)
    # Sorting by time makes the causal matrix upper-triangular
    order = np.argsort(t)
    # Return the (t, x) columns: these are the elements V of the causal set
    return np.column_stack([t[order], x[order]])
```

```
def causal_matrix(pts):
    # Number of elements n; the matrix will be n × n
    n = len(pts)
    # All time differences Δt = t_j − t_i at once
    dt = pts[None, :, 0] - pts[:, None, 0]
    # All spatial differences Δx = x_j − x_i likewise
    dx = pts[None, :, 1] - pts[:, None, 1]
    # i ≺ j iff j lies inside the future light cone of i (Δt > |Δx|)
    prec = (dt > np.abs(dx))
    # No element precedes itself (irreflexivity)
    np.fill_diagonal(prec, False)
    # Boolean matrix C[i, j] = (i ≺ j): the partial order that *is* time here
    return prec
```

```
def rank_longest_chain(C):
    # Number of elements
    n = C.shape[0]
    # Initialise each element's rank (length of the longest chain into its past) to 0
    rank = np.zeros(n, dtype=int)
    # Sweep in time order (C is upper-triangular, so the past has smaller indices)
    for j in range(n):
        # Collect the indices of every element in j's past
        past = np.where(C[:, j])[0]
        # Empty past means rank 0 (an element with no "birth")
        if len(past):
            # One more than the largest rank in the past: which tier of the procession this is
            rank[j] = rank[past].max() + 1
    # Return every element's rank
    return rank
```

```
def links(C):
    # Keep only links (direct causation): if C² is true there is an intermediary
    via = (C.astype(int) @ C.astype(int)) > 0
    # Relations with no intermediary are the links — the edges of the Hasse diagram
    return C & ~via
```

The length of the longest chain into the past is the rank: which tier of the procession an element stands on.

— 1.2 Emergent Dimension

As a touchstone of the principle that an element has nothing but its order, we extract the dimension from the order alone. Myrheim and Meyer showed that in a causal set of spacetime dimension d the fraction f of related pairs is a function of d only, $f(d) = \Gamma(d+1)\Gamma(d/2) / (2\Gamma(3d/2))$ (1/2 at d = 2, 1/10 at d = 4). Solving the observed f for d gives d ≈ 2.01 for the sprinkle of this paper (1+1 dimensions; Fig. 12-I). The elements carry no coordinates, yet a dimension emerges: the shape of spacetime is not an attribute of the elements but a statistic of their relations.

EMERGENT DIMENSION (MYRHEIM–MEYER ESTIMATOR)

```
def ordering_fraction(C):
    # Number of elements
    n = C.shape[0]
    # Related pairs divided by all pairs: the density of the order
    return float(C.sum()) / (n * (n - 1) / 2.0)
```

```
def myrheim_meyer_dimension(C):

# In a causal set of spacetime dimension d the fraction of related pairs is f(d) = Γ(d+1)Γ(d/2) / (2 Γ(3d/2)) (Myrheim–Meyer; 1/2 at d = 2, 1/10 at d = 4)
    f = ordering_fraction(C)
    # The formula as a function of d
    def expected(d):
        # Computed with the gamma function
        return math.gamma(d + 1) * math.gamma(d / 2) / (2 * math.gamma(1.5 * d))
    # Bisect d between 1 and 6 for expected(d) = f (f decreases monotonically in d)
    lo, hi = 1.0, 6.0
    # Forty steps give ample precision
    for _ in range(40):
        # Midpoint
        mid = 0.5 * (lo + hi)
        # If the expected fraction at the midpoint exceeds the observed one, the dimension is larger
        lo, hi = (mid, hi) if expected(mid) > f else (lo, mid)
    # The dimension estimated from order alone: elements carry no coordinates, yet a dimension emerges
    return 0.5 * (lo + hi)
```

— 1.3 Output Scheme

plate / paper PDF / note draft / SNS text. The plate is fixed as the primary artifact; the other outputs derive from the same Model/Controller state (same seed and parameter set). The v10 plate (Fig. 11) has four panels: I the sprinkled causal set and the horizon, II the hologram H, III pair correlation split by the horizon, IV the rank distribution.

SECTION 2

Horizon, Controller, and Counterexamples

Invariants are the skeleton that, once broken, leaves no artwork. We fix at least causal consistency, the two-phase behaviour across the horizon, stratification (rank, chains), and diversity (non-collapse of entropy). Making the counterexamples explicit fixes the forbidden regions, and gives the stability conditions a concrete meaning.

— 2.1 The Horizon as a Null Surface

The horizon B is placed as the surface on which one light-cone coordinate, u = (t − x)/√2, is constant: the local form of U = 0 in Kruskal coordinates, that is, of the Schwarzschild horizon. Causal relations never decrease u, so a thread can cross from outside to inside but never return. “Inside” means the region that has the horizon in its past.

SPLITTING BY THE HORIZON

```
def horizon_split(pts, u_h=0.5):
    # Recover the light-cone coordinate u = (t - x)/√2; the horizon is the null surface u = u_h (like U = 0 in Kruskal coordinates)
    u = (pts[:, 0] - pts[:, 1]) / math.sqrt(2.0)
    # Elements with u ≥ u_h are "inside"; causal relations never decrease u, so no thread ever returns from inside to outside
    inside = u ⩾ u_h
    # Return the inside mask; outside is its negation
    return inside
```

```
def cross_links(L, inside):
    # Extract the start and end indices of every link
    i, j = np.where(L)
    # Links whose endpoints differ in `inside` are the threads crossing the horizon
    crossing = inside[i] ≠ inside[j]
    # Return the count of crossing threads and their index pairs
    return int(crossing.sum()), list(zip(i[crossing], j[crossing]))
```

```
def horizon_area(L, inside):
    # Measure the horizon "area" as the number of links crossing it (Dou–Sorkin horizon entropy: area ∝ crossing links)
    return cross_links(L, inside)[0]
```

Following Dou and Sorkin, the horizon “area” is measured by the number of links crossing it.

— 2.2 The Controller θ and the Implementation of f

The four components of θ acquired their meaning empirically across v1–v9: p is the transmissivity of threads across the horizon, q the retention of threads that do not cross it, σ the positional jitter, and κ the fraction of random interference added across the pillars. f is nothing more than the successive application of these — jitter, pruning, interference.

```
THE CONTROLLER Θ AND THE PROJECTION RULE F
@dataclass
class Controller:
    # p: transmissivity / mixing — how freely threads pass the horizon (0, 1)
    p: float
    # q: selection / closure — how many non-crossing threads are retained (0, 1); q → 0 isolates
    q: float
    # σ: jitter — standard deviation of positional noise (≥ 0)
    sigma: float
    # κ: cross-pillar interference — fraction of connections cutting across pillars (≥ 0)
    kappa: float
```

```
def project(pts, L, inside, theta, seed=0):
    # Projection also needs randomness (pruning, jitter); fix the seed for reproducibility
    rng = np.random.default_rng(seed)
    # Output coordinates H = input coordinates + Gaussian jitter σ (the test of A1)
    H = pts + rng.normal(0.0, theta.sigma, size=pts.shape)
    # List both ends of every link
    i, j = np.where(L)
    # Keep crossing threads with probability p, others with probability q
    keep_p = np.where(inside[i] != inside[j], theta.p, theta.q)
    # Compare with uniform randomness to decide which threads survive
    keep = rng.random(len(i)) < keep_p
    # Collect the surviving threads as an (i, j) array
    E = np.column_stack([i[keep], j[keep]])
    # Add random long-range connections in proportion to κ (cross-pillar interference)
    m = int(theta.kappa * len(E))
    # Generate random pairs (indices are time-ordered, so enforce a < b for direction)
    extra = np.sort(rng.integers(0, len(pts), size=(m, 2)), axis=1)
    # Drop self-loops
    extra = extra[extra[:, 0] != extra[:, 1]]
    # Original threads plus interference threads form the edge set of the hologram H
    return H, np.vstack([E, extra]) if m else E
```

— 2.3 Observables

Checking the axioms requires quantities legible from H: edge entropy (Shannon entropy of the degree distribution), reachability depth (longest path), the number of threads crossing the horizon, and the pair correlation g(r), normalised by a uniform sprinkle of the same region. Sprinkling itself has no scale — correlation comes from the order, not from position — as panel III of Fig. 11 shows.

OBSERVABLES

```
def edge_entropy(E, n):
    # Count each element's degree (number of attached threads)
    deg = np.bincount(E.ravel(), minlength=n).astype(float)
    # Normalise degrees to a probability distribution (return 0 if there are no threads)
    if deg.sum() == 0:
        # Isolation: zero diversity
        return 0.0
    # Probabilities p_i = deg_i / Σdeg
    P = deg / deg.sum()
    # Shannon entropy −Σ p log p, skipping zero terms
    P = P[P > 0]
    # Return in nats; the maximum is log n (complete homogenisation)
    return float(-(P * np.log(P)).sum())
```

```
def reachability_depth(E, n):
    # Build a directed graph from the threads (time-ordered indices make it acyclic)
    G = nx.DiGraph()
    # Register every element as a vertex
    G.add_nodes_from(range(n))
    # Register the threads as edges
    G.add_edges_from(map(tuple, E))
    # Length of the longest path = reachability depth: how far the procession extends
    return nx.dag_longest_path_length(G)
```

```
def pair_correlation(H, ref, r_max=0.6, bins=36):
    # Compute distances for every pair (upper triangle only, excluding self-distances)
    d = np.sqrt(((H[:, None, :] - H[None, :, :]) ** 2).sum(axis=2))[np.triu_indices(len(H), k=1)]
    # Pair distances of the reference cloud (a uniform sprinkle of the same region) likewise
    d0 = np.sqrt(((ref[:, None, :] - ref[None, :, :]) ** 2).sum(axis=2))[np.triu_indices(len(ref),
k=1)]
    # Histogram of distances (pairs per r)
    hist, edges = np.histogram(d, bins=bins, range=(0.0, r_max))
    # Reference histogram: the denominator that cancels the bias of the region's shape
    hist0, _ = np.histogram(d0, bins=bins, range=(0.0, r_max))
    # Bin centres as r
    r = 0.5 * (edges[:-1] + edges[1:])
    # g(r) = observed / reference (normalised by total pairs); g ≈ 1 means no correlation, peaks mean structure
    g = (hist / max(hist.sum(), 1)) / np.maximum(hist0 / max(hist0.sum(), 1), 1e-9)
    # Return r and g(r)
    return r, g
```

— 2.4 Counterexamples

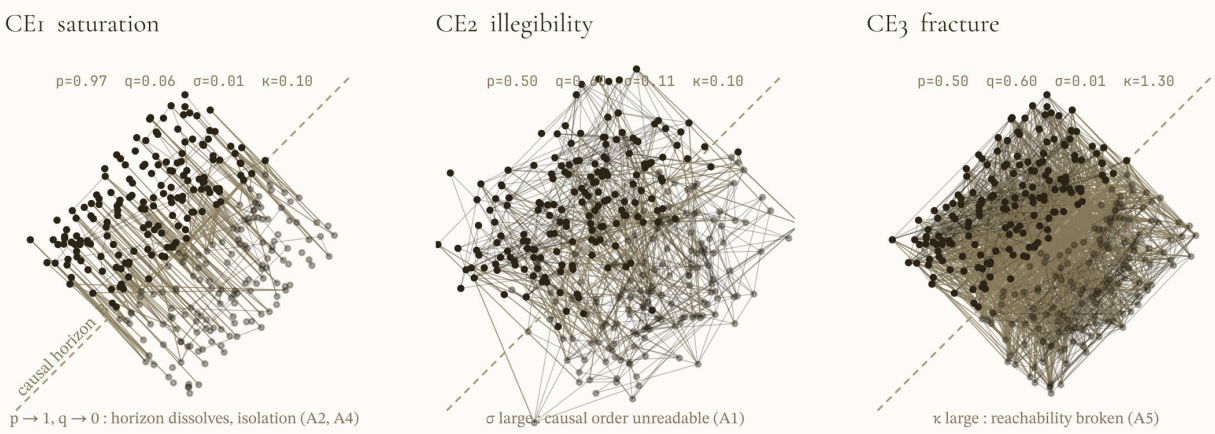

Fig. 2 Counterexamples (collapse modes) computed in the v10 model — CE1 saturation ($p \to 1$, $q \to 0$), CE2 illegibility (large σ), CE3 fracture (large κ).

The three collapses appear directly in the three panels of Fig. 2. In CE1 threads cross the horizon freely, inner threads vanish, and the two phases are lost. In CE2 the jitter exceeds the minimal link length, and one can no longer read which element follows which. In CE3 the threads crossing the horizon exceed its area, and the column fractures.

SECTION 3

Protocol Mapping (Production → Submission → Publication)

The View layer standardises four outputs: plate / paper PDF / note draft / SNS text. The plate is the primary artifact; the paper fixes the specification; the note provides the public narrative; the SNS text is a summary for announcement. Deriving all four from the same internal state keeps the process traceable and prevents divergence between the artwork and its language. In v10 the figures themselves are generated from the code in this paper, closing the gap between figure and specification as well.

SECTION 4

v9 — A Minimal Axiom System for the Projection Rule f

The projection rule f, fixed empirically across v1–v8.1, is reconstructed deductively from a minimal axiom system of five axioms (A1–A5). They hold simultaneously if and only if f can generate the structure we call a funeral procession — isolated cells bound by causal threads into a column. Each collapse counterexample (CE1–CE3) is reinterpreted as the violation of a specific axiom, and the stable regime is described algebraically as the set of parameters on which all five axioms hold at once.

4.1 The Known Structure, Restated

```
H = f(S; B, θ)
S = (V, ≺)                  causal set: elements V with causal order ≺
B                           causal horizon (null surface u = u_h)
θ = { p, q, σ, κ, H_edge }  Controller parameters
H                           View = Hologram (output)
```

4.2 The Five Minimal Axioms

A1 Order Preservation

```
a ≺ b in S   ⇒   f(a) precedes f(b) in H
```

f must preserve the partial order of S inside H. If time is not a flow but a depth of reachability, f must not destroy that ordering of depth.

Violated when σ is too large (CE2): jitter exceeds the minimal link length ≈ 1/√n, and one can no longer read which cell comes after which.

A2 Boundary Duality

```
B splits V into V_inside and V_outside;
f(V_inside) and f(V_outside) are structurally distinct.
```

As p → 1, inside and outside mix and the horizon disappears; without a horizon the procession is no longer a column but a point cloud. Conversely, if no thread crosses, the two worlds become unrelated, and again no column forms.

Violated when p→1, q→0 (CE1): saturation is the state in which the horizon loses meaning and diversity collapses at once.

A3 Local Continuity

```
for small δθ, the change δH is bounded and continuous;
θ lies in the interior of Ω_stable.
```

This is the axiom of reproducibility: the same seed and the same θ yield the same H — the foundation of the Protocol Mapping. At the boundary of the stable regime continuity breaks and a phase-transition-like collapse occurs. In v10 it is tested by requiring the other axioms to survive on the four neighbours of θ.

Violated at every boundary of CE1–CE3: the outer edge of the stable regime is a phase-transition point.

A4 Entropy Non-collapse

```
H_edge( f(S; B, θ) ) ≥ H_min > 0
```

As edge entropy tends to zero, f loses diversity. If all connections become equiprobable (homogenisation) or all vanish (isolation), the procession loses its meaning — the bonds of fate disappear. Edge entropy was introduced at v7 precisely to make this axiom visible.

Violated at CE1 (isolation as q→0).

A5 Holographic Bound

```
| cross-horizon threads |  ≤  Area(B) / ℓ_P²
```

The number of causal threads crossing the horizon has an upper bound proportional to the area of the horizon — a discrete analogue of the Bekenstein–Hawking boundary entropy. In v10 the area is measured, following Dou and Sorkin, by the number of links crossing the horizon, and the bound is tested by maximal flow (§5.2).

Violated when κ is too large (CE3): too many crossing threads break reachability and fracture the column.

— 4.3 Axioms and Collapse Counterexamples

Counterexample	Condition	Axiom violated	Phenomenon
CE1	p→1, q→0	A2, A4	horizon loss + isolation
CE2	large σ	A1	causal order illegible
CE3	large κ	A5	reachability fractured

— 4.4 The Stable Regime: Algebra and Phase Diagram

```
Ω_stable = { θ = (p, q, σ, κ) :
   A1:  σ < σ_max(n)          σ below the minimal link length
   A2:  r_min < ρ(θ) < r_max  crossing ratio ρ neither severed nor saturated
   A3:  θ ∈ int(Ω_stable)     interior only (the boundary is non-continuous)
   A4:  H_edge(θ) ≥ H_min     entropy floor
   A5:  κ < κ_max(p, q)       κ cap from the holographic bound  }
```

The v8.1 observation — stability at mid-range p/q, small σ, moderate κ — is obtained in v10 as an actual phase diagram (Fig. 3). The upper bound on p is set by A2 (saturation), the lower bound on q by A4 (isolation). The thresholds (r_min = 0.15, r_max = 0.85, H_min = 0.94 log n) are provisional values drawn from the v1–v9 experience and, as §10 states, should eventually be replaced by a derivation from the holographic boundary entropy.

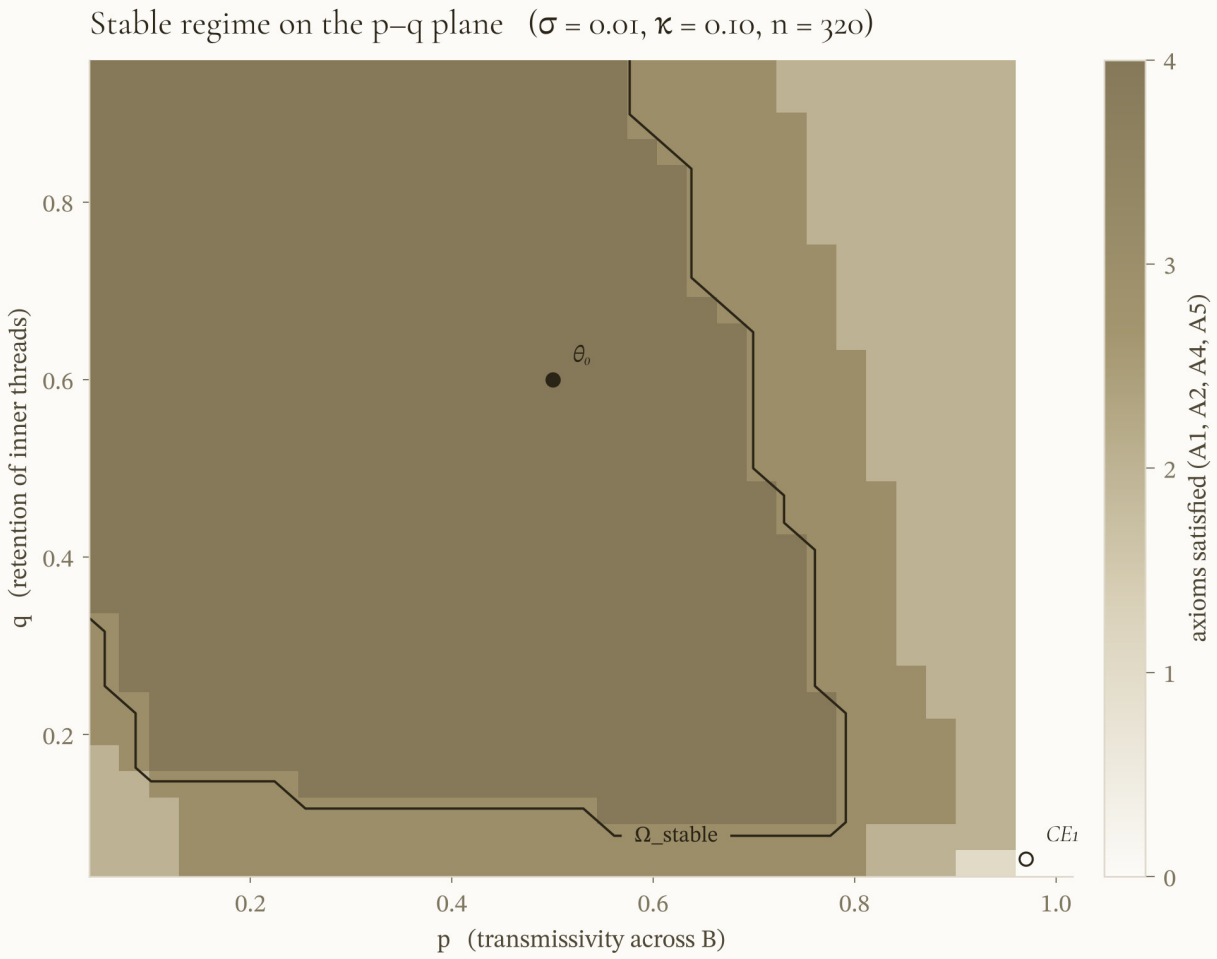

Fig. 3 Stable regime on the p–q plane (computed; σ = 0.01, κ = 0.10, n = 320). The dark region is Ω_stable; θ_0 = (0.5, 0.6) is the v10 plate setting; CE1 at the lower right is saturation.

— 4.5 Axiom-verification Functions

CHECKING THE FIVE AXIOMS

```
def check_axioms(theta, obs):
    # obs is a dict of observables: entropy, depth, cross, area, n
    p, q, s, k = theta.p, theta.q, theta.sigma, theta.kappa
    # A1 order preservation: jitter σ must stay below the minimal link length (≈ 1/√n)
    a1 = s < 0.5 / math.sqrt(obs["n"])
    # A2 boundary duality: crossing threads neither fade out (two severed worlds) nor saturate (horizon dissolves)
    a2 = 0.15 < obs["cross_ratio"] < 0.85
    # A4 entropy non-collapse: degree entropy stays above the floor H_min (94% of log n)
    a4 = obs["entropy"] > 0.94 * math.log(obs["n"])
    # A5 holographic bound: crossing threads ≤ horizon area (element count)
    a5 = obs["cross"] ⩽ obs["area"]
    # A3 local continuity is judged on a neighbourhood of θ, so it is delegated to is_interior
    return {"A1": a1, "A2": a2, "A4": a4, "A5": a5}
```

```
def observe(pts, L, inside, theta, seed=0):
    # Apply the projection rule f to obtain the hologram H and threads E
    H, E = project(pts, L, inside, theta, seed)
    # Number of elements
    n = len(pts)
    # Number of crossing threads (after projection)
    cross = int((inside[E[:, 0]] ≠ inside[E[:, 1]]).sum()) if len(E) else 0
    # Crossing threads before projection (denominator of the ratio)
    cross0 = max(cross_links(L, inside)[0], 1)
    # Bundle the observables in a dict
    return H, E, {"n": n, "entropy": edge_entropy(E, n), "depth": reachability_depth(E, n),
                  "cross": cross, "cross_ratio": cross / cross0, "area": horizon_area(L, inside)}
```

```
def is_interior(pts, L, inside, theta, eps=0.04):
    # A3: θ is interior if A1, A2, A4, A5 survive all four eps-perturbations
    for dp, dq in [(eps, 0), (-eps, 0), (0, eps), (0, -eps)]:
        # Build the neighbour θ′ (clamped to 0.01–0.99)
        t2 = Controller(min(max(theta.p + dp, 0.01), 0.99), min(max(theta.q + dq, 0.01), 0.99),
theta.sigma, theta.kappa)
        # Observe at θ′ and check the axioms
        ok = check_axioms(t2, observe(pts, L, inside, t2)[2])
        # If any axiom fails, θ sits on the boundary (a phase-transition point)
        if not all(ok.values()):
            # Not interior
            return False
    # All four directions held: interior of the stable regime
    return True
```

```
def is_stable(pts, L, inside, theta):
    # Check A1, A2, A4, A5 at θ itself
    ok = check_axioms(theta, observe(pts, L, inside, theta)[2])
    # Check A3 on the neighbourhood
    ok["A3"] = is_interior(pts, L, inside, theta)
    # Only when all five are true does θ belong to Ω_stable
    return all(ok.values()), ok
```

COMPUTING THE PHASE DIAGRAM

```
def phase_diagram(pts, L, inside, sigma=0.01, kappa=0.1, grid=25):
    # Sweep p and q on a grid over (0, 1)
    ps = np.linspace(0.04, 0.96, grid)
    # Same grid for q
    qs = np.linspace(0.04, 0.96, grid)
    # Record the number of satisfied axioms at each grid point
    S = np.zeros((grid, grid))
    # Also record entropy at each point (used for contour lines)
    Hn = np.zeros((grid, grid))
    # Double loop over the grid
    for a, q in enumerate(qs):
        # Inner loop over p
        for b, p in enumerate(ps):
            # θ at this grid point
            th = Controller(p, q, sigma, kappa)
            # Observe and check axioms (A3 is costly, so it is omitted in the sweep)
            _, _, obs = observe(pts, L, inside, th)
            # Count of true axioms (0–4)
            S[a, b] = sum(check_axioms(th, obs).values())
            # Entropy normalised by log n to 0–1
            Hn[a, b] = obs["entropy"] / math.log(len(pts))
    # Return the grid and both fields
    return ps, qs, S, Hn
```

— 4.6 On the Uniqueness of f

Theorem (informal): within Ω_stable, a mapping f satisfying A1–A5 is unique up to isomorphism — the claim that the structure called a funeral procession is uniquely determined by these constraints. Physically: in the holographic principle the bulk is uniquely reconstructed from boundary data. In Buddhist terms: the phenomenal world unfolds uniquely from the storehouse consciousness. Artistically: once seed and θ are fixed, the plate is uniquely determined — this is what a reproducible production process means.

Sketch of proof: (1) assume f satisfies A1–A5; (2) for any S, B, θ ∈ Ω_stable compute the structural observables of f(S; B, θ) — reachability depth, edge entropy, g(r); (3) show these depend only on θ and not on the choice of S (universality); (4) show that if two mappings f, f′ share the same observables, there is an isomorphism φ with f′ = φ ∘ f. In v10 step (2) became computable. Steps (3) and (4) remain unproved and are left as the open problems of §10.

SECTION 5

v10 — Inside the Horizon

— 5.1 The Penrose Diagram and the Event Horizon

In the Penrose diagram of maximally extended Schwarzschild spacetime, the exterior region I and the interior II are divided by the 45-degree null line B (Fig. 4). B is a light path; along B no proper time passes. An interior worldline, seen from outside, slows asymptotically toward B; seen from inside, it heads for the singularity r = 0. This paper treats a B for which the interior worldlines are described only while they remain within the domain of f (Ω_stable): an idealisation of a static Schwarzschild horizon, with evaporation (Hawking radiation) and the singularity left out of scope. The singularity is where f is undefined, and this paper claims nothing about it.

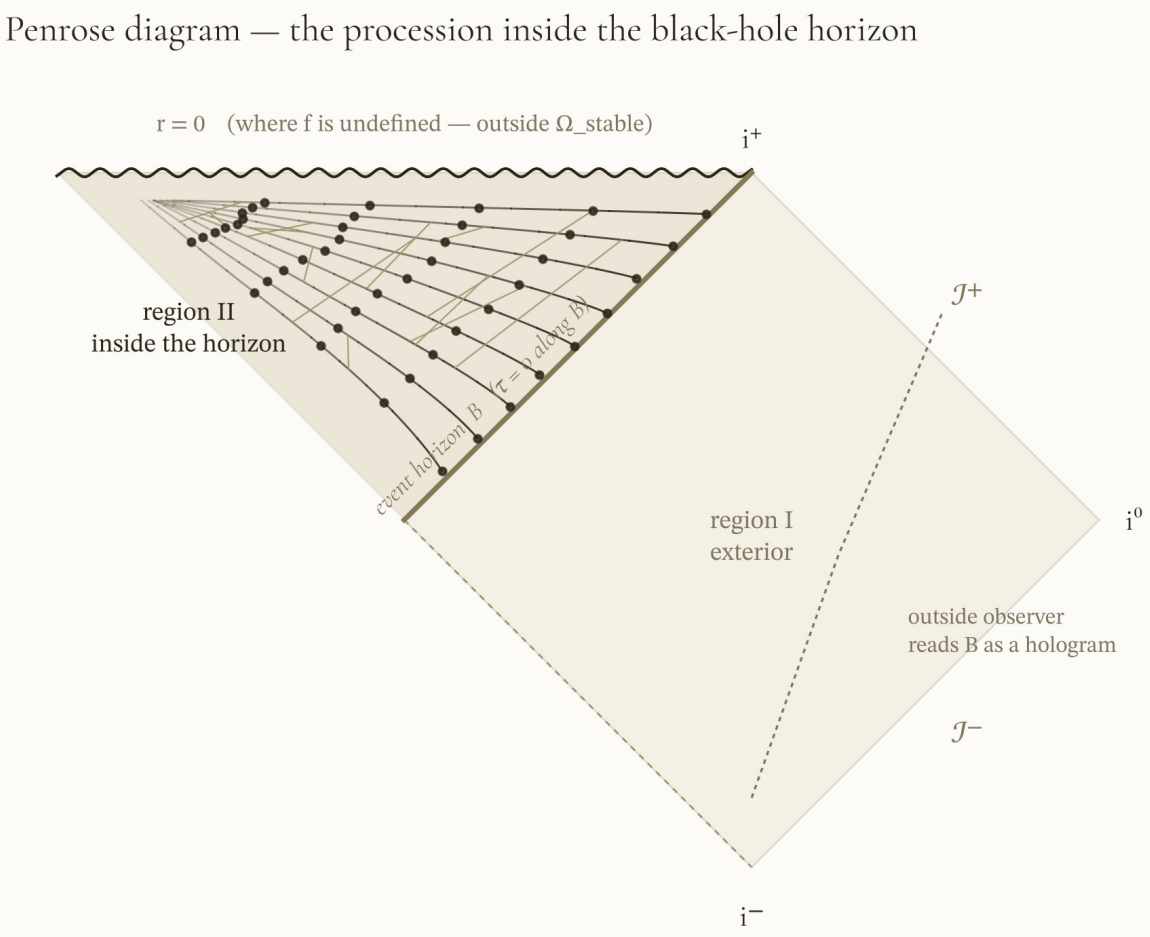

Fig. 4 Penrose diagram — the procession inside the black-hole horizon. The outside observer reads B as a hologram.

This is the content of thesis P2. What we call “now” is a set of elements bound to one another by threads, inside the region that has the horizon in its past. For the outside observer, all our events are read as information projected onto B. For us, B is an uncrossable past. These are two readings of one structure; neither is real at the expense of the other.

— 5.2 Bit Threads and the Derivation of κ_max

Freedman and Headrick rephrased the holographic entropy formula (Ryu–Takayanagi) not in terms of a minimal surface but of maximal flow: the entropy of a boundary region is the maximal number of independent threads (bit threads) that can leave it, and by the max-flow min-cut theorem this equals the area of the smallest surface that cuts them — the horizon. A5 in this paper is the discrete version of that rephrasing.

DERIVING K_MAX VIA BIT THREADS

```
def max_threads_across(E, inside, n):
    # Build an undirected graph (a thread can carry flow either way)
    G = nx.Graph()
    # All elements as vertices
    G.add_nodes_from(range(n))
    # Give every thread capacity 1: one thread carries exactly one bit
    for a, b in E:
        # Read the capacity of an existing edge (0 if none)
        c0 = G.get_edge_data(int(a), int(b), {"capacity": 0})["capacity"]
        # Increase the capacity by 1 (bundle parallel threads)
        G.add_edge(int(a), int(b), capacity=c0 + 1)
    # Super-source S to every inside element, super-sink T to every outside element, infinite capacity
    S, T = n, n + 1
    # Register the super-vertices first (so T exists even if the outside is empty)
    G.add_nodes_from([S, T])
    # To inside
    G.add_edges_from((S, int(v), {"capacity": float("inf")}) for v in np.where(inside)[0])
    # To outside
    G.add_edges_from((int(v), T, {"capacity": float("inf")}) for v in np.where(~inside)[0])
    # Max flow = bits simultaneously carried inside→outside = min cut (the horizon "area")
    flow, _ = nx.maximum_flow(G, S, T)
    # Return as an integer
    return int(flow)
```

```
def kappa_max(pts, L, inside, p, q, sigma=0.01, trials=12):
    # Increase κ from 0 and find the smallest κ at which A5 (crossing ≤ area) fails
    area = horizon_area(L, inside)
    # Candidate κ values
    for k in np.linspace(0.0, 1.5, trials):
        # Observe at this κ
        _, E, obs = observe(pts, L, inside, Controller(p, q, sigma, k))
        # Count threads that truly pass via max flow (independent, not merely crossing)
        if max_threads_across(E, inside, len(pts)) > area:
            # Area exceeded: this is κ_max
            return float(k)
    # No violation within the sweep
    return 1.5
```

κ_max(p, q) is fixed numerically as the smallest κ at which the maximal flow across the horizon exceeds its area.

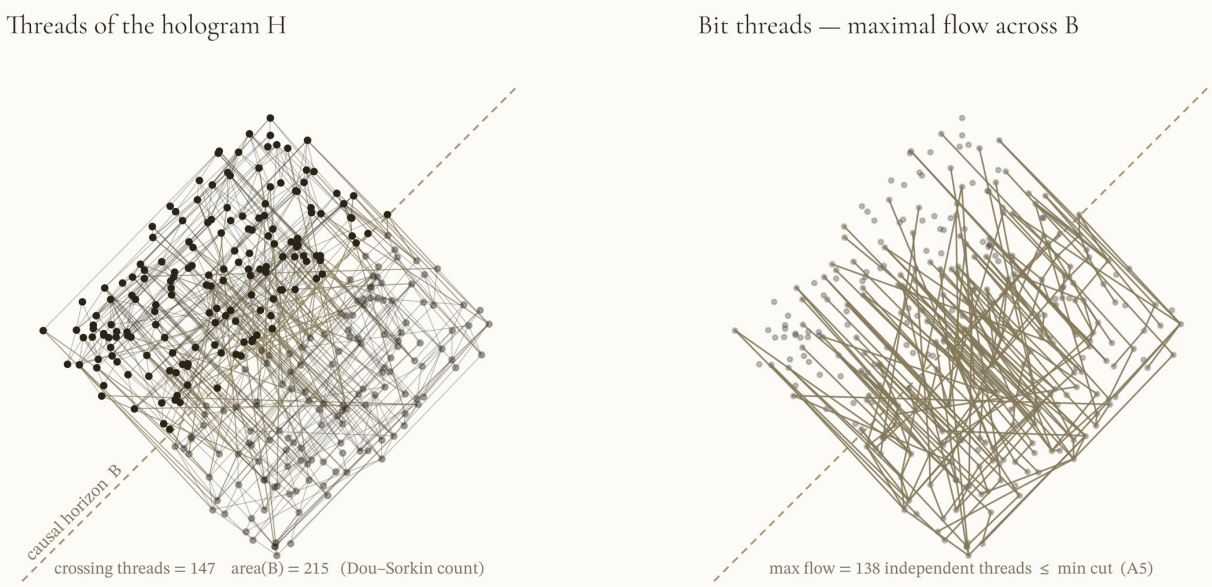

Fig. 5 Threads of the hologram H (left) and bit threads as the maximal flow across the horizon (right). The maximal flow never exceeds the minimum cut, i.e. the area.

v9 wrote κ_max only as "an upper bound depending on p, q." In v10, at $\theta_0 = (0.5, 0.6)$ we find κ_max ≈ 0.27; lowering p to (0.3, 0.5) gives κ_max ≈ 0.41. The lower the transmissivity p, the more of the horizon's area remains available for interference — consistent with reading the Bekenstein bound as permitting whatever of the area is not already in use.

— 5.3 The Dou–Sorkin Horizon Entropy

Dou and Sorkin proposed counting the entropy of a black hole in a causal set by the number of links crossing the horizon. This paper measures the horizon area by that count, so that A5 takes the nearly trivial form "crossing threads cannot exceed the links counted as area." That near-triviality matters: an axiom should be as obvious as possible.

— 5.4 Testing the Area Law — the Naive Count Fails

The physical ground of A5 is the area law: if the threads crossing the horizon scale with its area rather than with the volume (the element count n), then in 1+1 dimensions they should scale as a length, $\propto \sqrt{n}$. Counting crossing links for n from 80 to 1280 gives a log–log slope of 1.04 — proportional to the volume, not the area (Fig. 12-II). The reason is peculiar to 1+1-dimensional causal sets: links are long and nearly null, and do not concentrate near the horizon. Dou and Sorkin obtained the area law through a refinement — counting, across a spacelike hypersurface Σ, only pairs maximal before Σ and minimal after it — which the naive count of this paper lacks. This negative result shows that two of the open problems of §10, the analytic derivation of the thresholds and the extension to 3+1 dimensions, are one and the same problem.

```
TESTING THE AREA LAW
def area_law(ns=(80, 160, 320, 640, 1280), seed=7):
    # Count the links crossing the horizon as the number of elements n varies
    out = []
    # For each n
    for n in ns:
        # Sprinkle, order, links, horizon
        pts = sprinkle_diamond(n, seed); L = links(causal_matrix(pts)); inside = horizon_split(pts)
        # Record the crossing count
        out.append((n, cross_links(L, inside)[0]))
    # In 1+1 dimensions the horizon "area" is a length ∝ √n; crossing links growing as n^0.5 would be an area law, as n^1 a volume law
    xs = np.log([a for a, _ in out]); ys = np.log([b for _, b in out])
    # Log–log slope (least squares)
    slope = float(np.polyfit(xs, ys, 1)[0])
    # Return the measurements and the slope. The naive count gives a slope near 1: links in 1+1 dimensions are long and nearly null, and do not concentrate near the horizon
    return out, slope
```

SECTION 6

Strings and Branes — A Reading from Superstring and M-theory

The next two sections state correspondences, not theorems. A correspondence is the observation that two structures share a shape, not the claim that they are the same thing. Where computation ends and reading begins is stated at the end of each subsection.

— 6.1 The Thread Is a String

The causal thread of this paper is a one-dimensional object joining two elements. The basic object of string theory is likewise a one-dimensional string, whose worldsheet traces a two-dimensional surface in spacetime. When a bit thread pierces the boundary as a single uncut strand, it is a cross-section of a string worldsheet. The maximal flow of §5.2 can be read as the maximal number of strings piercing the horizon.

— 6.2 T-duality and a World Without Size

Wrap a closed string on a circle of radius R: the mass spectrum is fixed by the momentum number n and the winding number w, and replacing R by α'/R merely exchanges n and w, leaving the spectrum unchanged. This is T-duality: a large circle and a small one cannot be told apart. In the context of this paper it is the string-theoretic form of thesis P1 — the size of the window (R) is not physics. As beginning and end are products of the window's edge, so large and small are only a relabelling of winding.

STRING DUALITY: T-DUALITY AND THE MASS SPECTRUM

```
def closed_string_spectrum(R, alpha=1.0, nmax=3, wmax=3):
    # Mass² of a closed string on a circle of radius R = (n/R)² + (wR/α')² + oscillators (omitted here)
    spec = set()
    # Momentum quantum number n (waves around the circle)
    for n in range(-nmax, nmax + 1):
        # Winding number w (how many times the string wraps the circle)
        for w in range(-wmax, wmax + 1):
            # Round mass² to 6 decimals and add to the set
            spec.add(round((n / R) ** 2 + (w * R / alpha) ** 2, 6))
    # Return the mass² set as a sorted tuple
    return tuple(sorted(spec))
```

```
def t_duality_holds(R, alpha=1.0):
    # T-duality: radius R and radius α'/R give the same spectrum (n and w exchange roles)
    return closed_string_spectrum(R, alpha) == closed_string_spectrum(alpha / R, alpha)
```

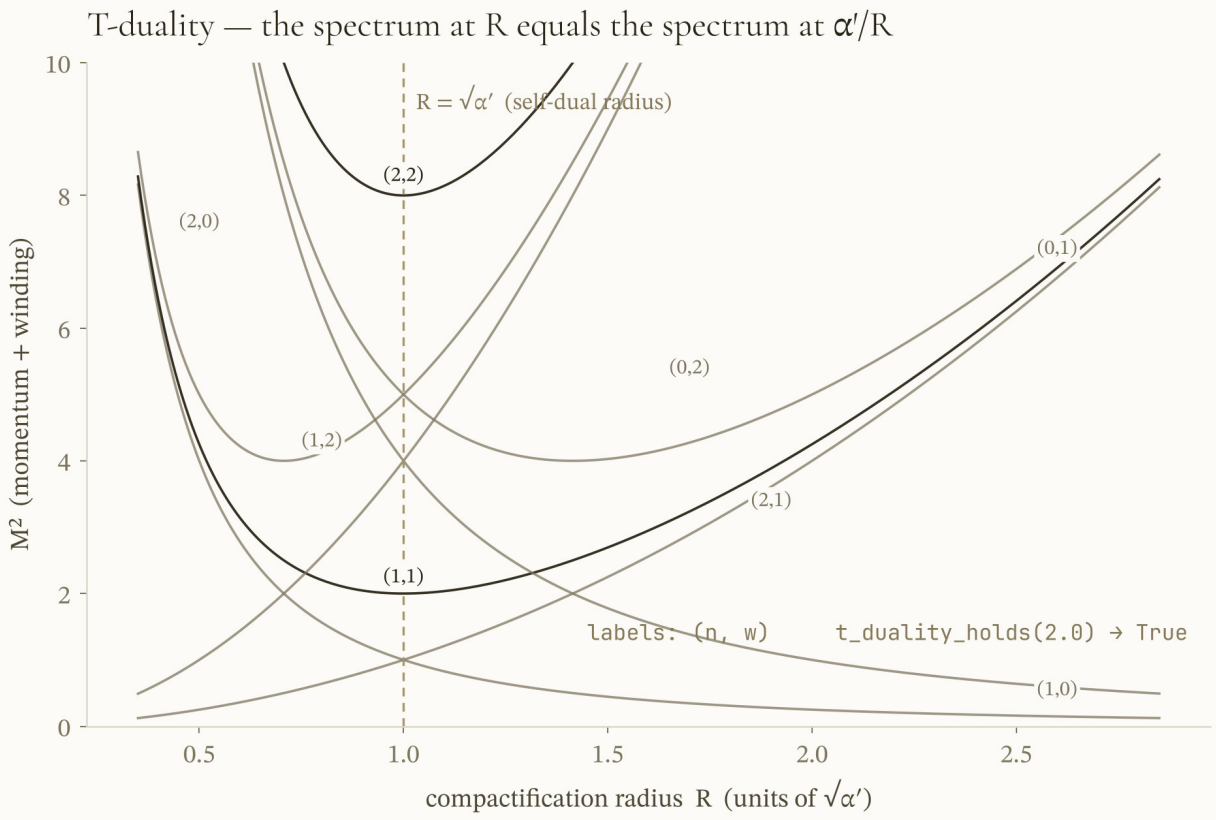

Fig. 7 T-duality — the spectrum at R equals the spectrum at α′/R. About the self-dual radius R = √α′, (n, w) and (w, n) are mirror images.

— 6.3 M-theory and the World on a Brane

The five superstring theories were unified as different limits of eleven-dimensional M-theory, where branes as well as strings become fundamental and the ends of open strings are fixed to branes. Brane cosmology takes our four-dimensional world to be a brane floating in a higher-dimensional spacetime, and the AdS/CFT correspondence showed that a theory on the boundary describes the gravity of the bulk completely. The horizon B of this paper is a brane in this sense. We do not live on the brane; we are inside, with B in our past, and we are projected onto B. Of §§6.1–6.3 only the T-duality formula is computed; the rest is correspondence.

This paper	String / M-theory	Status
causal thread	cross-section of a string worldsheet	correspondence
maximal flow of bit threads	number of strings piercing the horizon	correspondence
horizon B	brane / AdS boundary	correspondence
window size is not physics (P1)	T-duality R ↔ α′/R	computed (§6.2)
A5 area law	Bekenstein–Hawking, Ryu–Takayanagi	discrete analogue

SECTION 7

Mandala and the Tree of Life — Concentric Projection of an Order

— 7.1 The Tree of Life Is a Partial Order

The kabbalistic Tree of Life consists of ten sephiroth and twenty-two paths. Give the paths the direction of emanation (Kether to Malkuth) and the tree becomes a directed acyclic graph — a partial order. Computing each sephira's rank, the length of the longest path from Kether, yields nine tiers from 0 to 8 (Fig. 6, left), and only the links — paths without an intermediary — remain as the thick strokes. What has been called the "Sephirot constraint" since v2 was the operation of laying this order over the causal set. Only in v10 can its content be written down: a homomorphism onto the ten-element partial order.

THE ORDER STRUCTURE OF THE TREE OF LIFE

```
SEPHIROTH = ["Kether", "Chokmah", "Binah", "Chesed", "Geburah", "Tiphereth", "Netzach", "Hod",
"Yesod", "Malkuth"]
# The 22 paths; each pair (a, b) is written in the direction of emanation (a flows into b)
PATHS = [(0, 1), (0, 2), (0, 5), (1, 2), (1, 3), (1, 5), (2, 4), (2, 5), (3, 4), (3, 5), (3, 6),
         (4, 5), (4, 7), (5, 6), (5, 7), (5, 8), (6, 7), (6, 8), (6, 9), (7, 8), (7, 9), (8, 9)]
```

```
def tree_of_life():
    # Build the Tree of Life as a directed graph
    G = nx.DiGraph(PATHS)
    # If acyclic, it is an order (emanation never flows back)
    acyclic = nx.is_directed_acyclic_graph(G)
    # Each sephira's rank = length of the longest path from Kether
    rank = {v: nx.dag_longest_path_length(G.subgraph(nx.ancestors(G, v) | {v})) for v in G}
    # Keep only links (paths without an intermediary) from the transitive closure
    TC = nx.transitive_closure_dag(G)
    # Path (a, b) is a link iff no c exists with a→c→b
    link = [(a, b) for a, b in PATHS if not any((a, c) in TC.edges and (c, b) in TC.edges for c in
G if c not in (a, b))]
    # Return the results
    return acyclic, rank, link
```

— 7.2 The Mandala Projection

A mandala is a concentric structure opening from the centre outward. Project the causal set with rank as radius and the spatial coordinate x wrapped onto the circle (a compactification — the same operation as the circle of §6.2): elements without a past gather at the centre and the deepest tier lies on the rim (Fig. 6, right), and the causal threads become radial lines. The Womb-realm mandala, with Mahāvairocana seated at the centre and the deities unfolding outward, has the same shape as this projection. The centre is not an uncaused cause; it is the set of elements without a past inside the window — that is, the window's edge (§8.1). The centre of the mandala is a point drawn by the limit of observation.

MANDALA PROJECTION: RANK ONTO CONCENTRIC RINGS

```
def mandala_layout(pts, rank):
    # Maximal rank: this becomes the outermost ring
    rmax = max(int(rank.max()), 1)
    # Radius = rank / rmax; rank 0 (elements with no past) at the centre, the deepest tier on the rim
    radius = rank / rmax
    # Wrap the spatial coordinate x onto the circle: map x ∈ [-a, a] to angle [0, 2π) (compactification)
    a = np.abs(pts[:, 1]).max() + 1e-9
    # Angle θ = π (x + a) / a
    theta = math.pi * (pts[:, 1] + a) / a
    # Polar to Cartesian; causal threads become radial lines running outward from the centre
    return np.column_stack([radius * np.cos(theta), radius * np.sin(theta)])
```

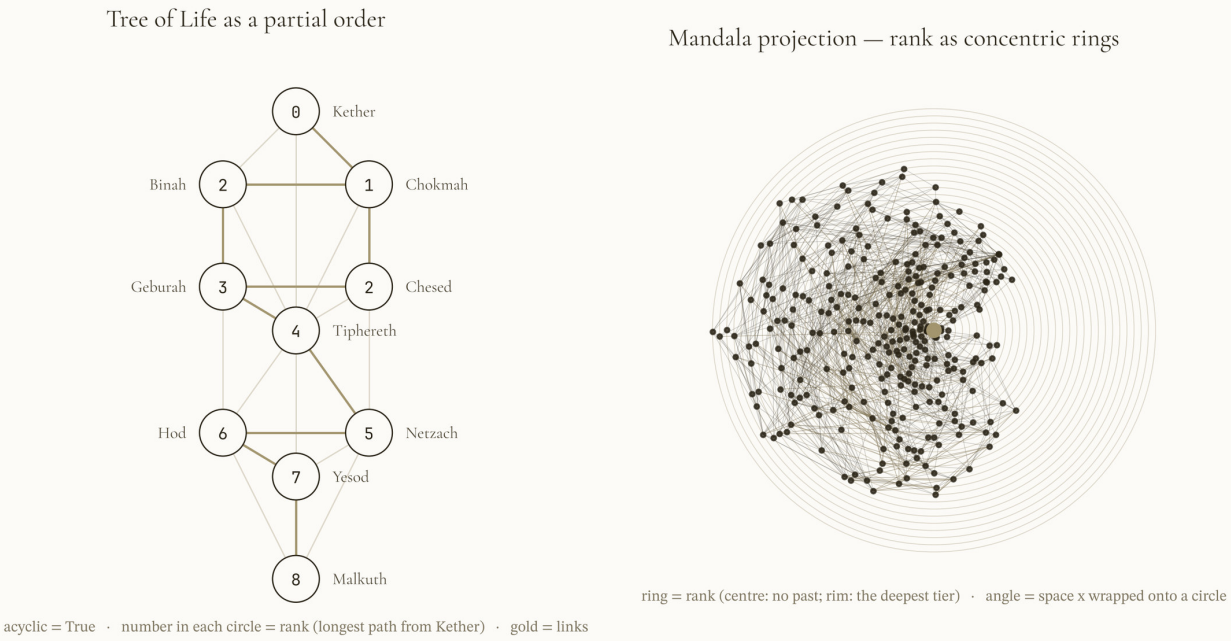

Fig. 6 Left — the Tree of Life as a partial order (numbers are ranks, gold strokes are links). Right — the mandala projection (rings are rank, angle is space x wrapped onto the circle).

7.3 The Two Realms and Boundary Duality

Shingon esoteric Buddhism pairs two mandalas, the Womb realm and the Diamond realm. A2 of this paper — that the inside and outside of the horizon remain two structurally distinct phases — has the form of one structure with two indispensable readings. The two readings of §5.1 — the outside observer reads B as a hologram; we inside hold B as an uncrossable past — have the same form. This is a correspondence, not a theorem.

SECTION 8

Unborn, Undying — Nirvāṇa Restated in Order Theory

8.1 Beginning and End Are the Window's Edge

A finite causal set always has minimal elements (no past) and maximal elements (no future), and it is natural to call them beginnings and endings. But if the observation window is held fixed while only the sprinkled region is enlarged, the fraction of window elements without a past falls rapidly to zero (Fig. 8). Enlarging the region by a factor of 1.2 is enough for the “born” elements to disappear. Beginning and end are properties not of the order but of the window that cuts it.

TESTING "UNBORN AND UNENDING"

```
def unborn_unending(C):
    # Does each element have a past (any true in its column)?
    has_past = C.any(axis=0)
    # Does each element have a future (any true in its row)?
    has_future = C.any(axis=1)
    # "Unborn" = has a past (not a first element); "unending" = has a future
    return has_past, has_future
```

```
def null_proper_time(pts_on_horizon):
    # Proper time between two horizon points τ² = Δt² - Δx²; on a null line it is always 0
    d = np.diff(pts_on_horizon, axis=0)
    # τ² per segment (floating error of order ±1e-12)
    tau2 = d[:, 0] ** 2 - d[:, 1] ** 2
    # Sum over all segments: the numerical statement that no time passes on the horizon
    return float(tau2.sum())
```

```python
def window_test(scales=(1.0, 1.02, 1.05, 1.1, 1.2, 1.35, 1.5, 2.0), density=320, seed=11):
    # Fix the observation window (unit diamond) and enlarge only the sprinkled region by `scale`
    out = []
    # Try each scale
    for s in scales:
        # Keep density constant: element count scales with area (scale²)
        pts = sprinkle_diamond(int(density * s * s), seed) * s
        # Translate so the enlarged region is centred on the window
        pts[:, 0] -= (s - 1.0) * math.sqrt(2.0) / 2.0
        # Causal matrix over the whole enlarged region
        C = causal_matrix(pts)
        # Elements inside the window (the original unit diamond: 0 ≤ u, v < 1)
        u = (pts[:, 0] - pts[:, 1]) / math.sqrt(2.0)
        # v likewise
        v = (pts[:, 0] + pts[:, 1]) / math.sqrt(2.0)
        # Window membership
        win = (u >= 0) & (u < 1) & (v >= 0) & (v < 1)
        # Fraction of window elements that have no past within the whole enlarged region
        no_past = float((~C[:, win].any(axis=0)).mean())
        # Likewise, the fraction with no future
        no_future = float((~C[win, :].any(axis=1)).mean())
        # Record (scale, beginning fraction, ending fraction)
        out.append((s, no_past, no_future))
    # Both tend to 0 as scale grows: beginning and end are products of the window's edge
    return out
```

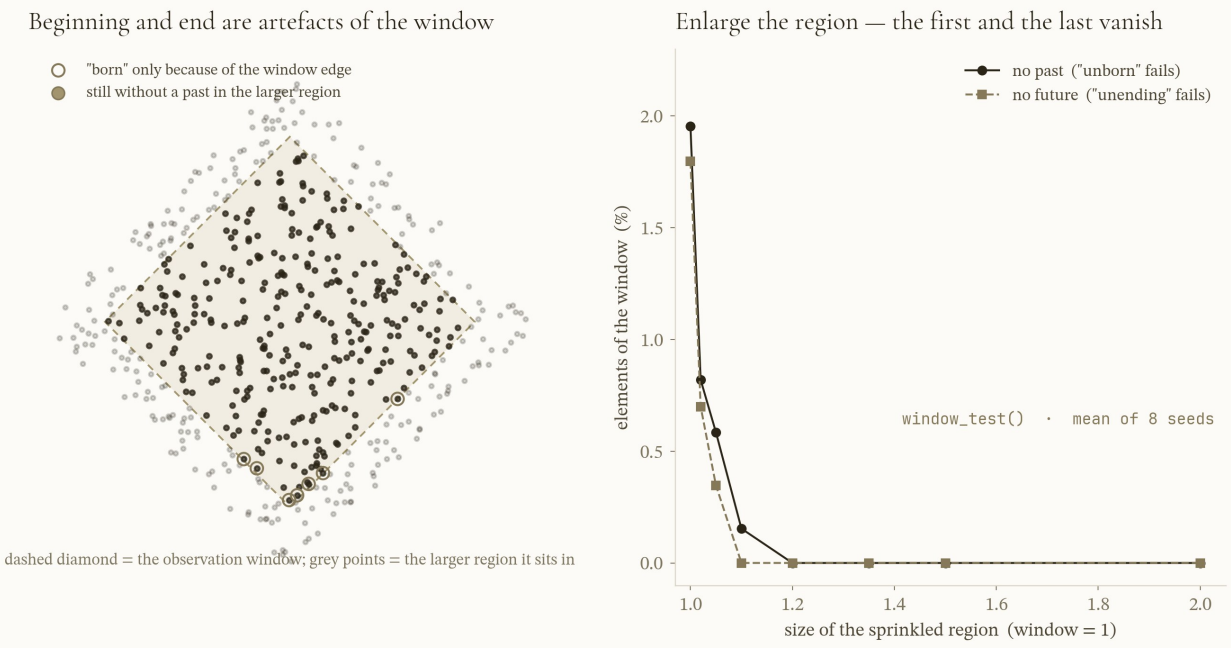

Fig. 8 Left — elements that look “born” at the window's edge (dashed diamond) have a past in the larger region. Right — enlarging the region sends the fractions without past and without future to zero (mean of 8 seeds).

— 8.2 No Time Passes on the Horizon

The horizon is a null surface: the proper time between two of its points is $\tau^2 = \Delta t^2 - \Delta x^2 = 0$. For a viewpoint along the horizon, every event inside is simultaneous. The words of the artwork — events of the past, present, and future unfold in an instant — are the very definition of the viewpoint on the horizon. Instead of calling nirvāṇa "outside time," this paper calls it the viewpoint on the surface where $\tau = 0$.

— 8.3 Dependent Origination, Non-self, Unborn-Undying

An element of a causal set has no attribute other than its order: what an element is, is exhausted by what it follows and what it precedes. This is a formal restatement of non-self (no element has an intrinsic substance), and dependent origination (everything arises from conditions) is the order $\prec$ itself. The Heart Sūtra's "neither born nor extinguished" corresponds to the window-edge theorem of §8.1. This paper does not define the nirvāṇa the Buddha reached. What it can define is that beginning and end are products of the window, that no time passes on the horizon, and that every element belongs to a column (§8.4). Beyond that lies practice, not computation.

— 8.4 Every One of Us Forms a Procession

Dilworth's theorem: a finite partial order can be partitioned into as many chains as the size of its largest antichain (its width). Fulkerson's proof uses a maximum matching in a bipartite graph — draw an edge wherever j may follow i, assign a successor to as many elements as possible, and the number of elements left without a successor is the number of chains. In the 320-element sprinkle, 33 columns cover every element (Fig. 10). Each element belongs to exactly one column; no one stands outside. Thesis P3 is a reading of this theorem.

DECOMPOSITION INTO PROCESSIONS (DILWORTH'S THEOREM)

```
def minimum_chain_cover(C):
    # Number of elements
    n = C.shape[0]
    # Build a bipartite graph: each element's "exit" on the left, its "entrance" on the right
    G = nx.Graph()
    # Register both sides (left 0..n−1, right n..2n−1)
    G.add_nodes_from(range(2 * n))
    # If i ≺ j, connect i's exit to j's entrance (j may follow i in a chain)
    i, j = np.where(C)
    # Register the edges in bulk
    G.add_edges_from(zip(i, j + n))
    # Maximum matching = the largest number of elements that are followed by someone
    M = nx.bipartite.maximum_matching(G, top_nodes=range(n))
    # Number of matched edges (the dict stores both directions, so halve it)
    matched = sum(1 for a in M if a < n)
    # Dilworth / Fulkerson: minimum chain cover = n − maximum matching; this is the number of processions
    return n - matched, M
```

```
def chains_from_matching(C, M):
    # Number of elements
    n = C.shape[0]
    # Record which elements follow someone (matched on the right side)
    follows = {M[b] : b - n for b in M if b ⩾ n}
    # Elements that follow no one are the heads of the processions
    heads = [v for v in range(n) if v not in follows.values()]
    # From each head, follow the matching to the next element to build the chain
    chains = []
    # For every head
    for h in heads:
        # The chain starts at the head
        chain = [h]
        # Advance while the exit is matched
        while chain[-1] in M and M[chain[-1]] ⩾ n:
            # The next element (map the right-side index back)
            chain.append(M[chain[-1]] - n)
        # Add the finished chain
        chains.append(chain)
    # Every element lies in exactly one chain: no one stands outside the procession
    return chains
```

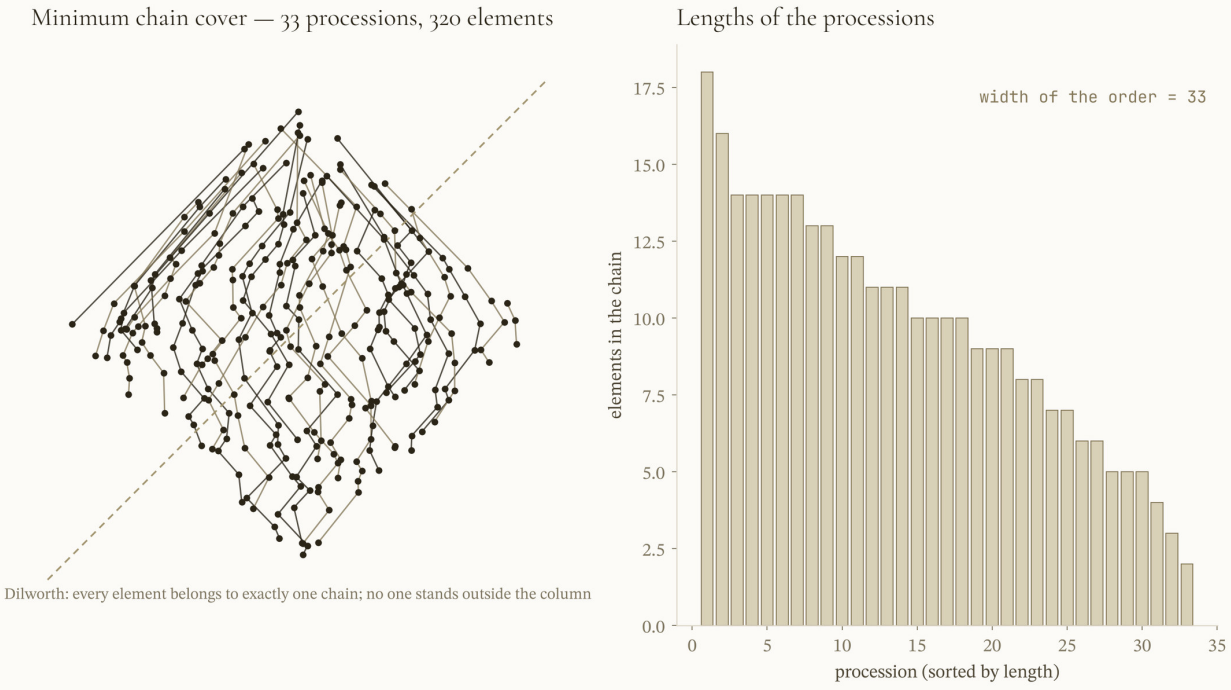

Fig. 10 Minimum chain cover — 320 elements fall into 33 processions (left); the length of each column (right).

— 8.5 The Present Is an Antichain

The number of processions has a dual. Dilworth's theorem states that the minimum number of chains equals the size of the maximum antichain — a set of mutually unrelated elements, none in the past or future of another: elements that are "at once." Relativity has no absolute surface of simultaneity, but a causal set always has

maximal sets of mutually unrelated elements, and these are the order-theoretic definition of the present. In the 320-element sprinkle the maximum antichain has 33 elements — exactly the number of processions (Fig. 12-III). The number of beings that can exist at once equals the number of columns: one present for each procession.

THE PRESENT IS AN ANTICHAIN

```
def maximum_antichain(C, M):
    # Number of elements
    n = C.shape[0]
    # Rebuild the same bipartite graph as for the chain cover (left = exits, right = entrances)
    G = nx.Graph(); G.add_nodes_from(range(2 * n)); i, j = np.where(C); G.add_edges_from(zip(i, j + n))
    # König's theorem: a minimum vertex cover follows from the maximum matching
    cover = nx.bipartite.to_vertex_cover(G, M, top_nodes=range(n))
    # Elements whose exit and entrance are both outside the cover form a maximum antichain (Dilworth's dual)
    anti = [v for v in range(n) if v not in cover and (v + n) not in cover]
    # An antichain is a set of mutually unrelated elements: any two are "simultaneous", and its size equals the number of processions
    return len(anti), anti
```

— 8.6 En — a Shared Future

In the artwork, three figures are joined through a single rose. In order-theoretic terms the rose is a shared future — an element k in the future of both i and j. Define en (a bond) as the existence of such a k, and it is computed at once by the matrix product $C\,C^{T}$. Within the window alone 93.9% of pairs share a future; with the window fixed and the region enlarged 1.2-fold the figure is 99.9%, and at 1.5-fold it reaches 100% (Fig. 12-IV). In a sufficiently large world no bond is ever cut: any two beings have a shared future.

EN: A SHARED FUTURE

```
def common_future_fraction(C):
    # Number of elements
    n = C.shape[0]
    # F[i, j] = (i and j share a future) = (some k has i ≺ k and j ≺ k)
    Ci = C.astype(int)
    # One matrix product: (C C^T)[i, j] = number of shared future elements of i and j
    shared = (Ci @ Ci.T) > 0
    # Count only pairs with i < j
    iu = np.triu_indices(n, k=1)
    # Fraction of pairs that share a future
    return float(shared[iu].mean())
```

```
def en_window_test(scales=(1.0, 1.2, 1.5, 2.0, 3.0), density=320, seed=11):
    # Fix the window, enlarge the region, and measure how many window pairs share a future (within the whole region)
    out = []
    # Each scale
    for s in scales:
        # Sprinkle the enlarged region, centred on the window
        pts = sprinkle_diamond(int(density * s * s), seed) * s; pts[:, 0] -= (s - 1.0) *
math.sqrt(2.0) / 2.0
        # Causal matrix
        C = causal_matrix(pts)
        # Window membership
        u = (pts[:, 0] - pts[:, 1]) / math.sqrt(2.0); v = (pts[:, 0] + pts[:, 1]) / math.sqrt(2.0)
        # Inside the window
        win = np.where((u ⩾ 0) & (u < 1) & (v ⩾ 0) & (v < 1))[0]
        # For window elements i, j, is there a shared future k anywhere in the region?
        Ci = C[win].astype(int)
        # Shared-future indicator
        shared = (Ci @ Ci.T) > 0
        # Upper-triangle fraction
        iu = np.triu_indices(len(win), k=1)
        # Record
        out.append((s, float(shared[iu].mean())))
    # As the region grows, every pair comes to share a future: no bond is ever cut
    return out
```

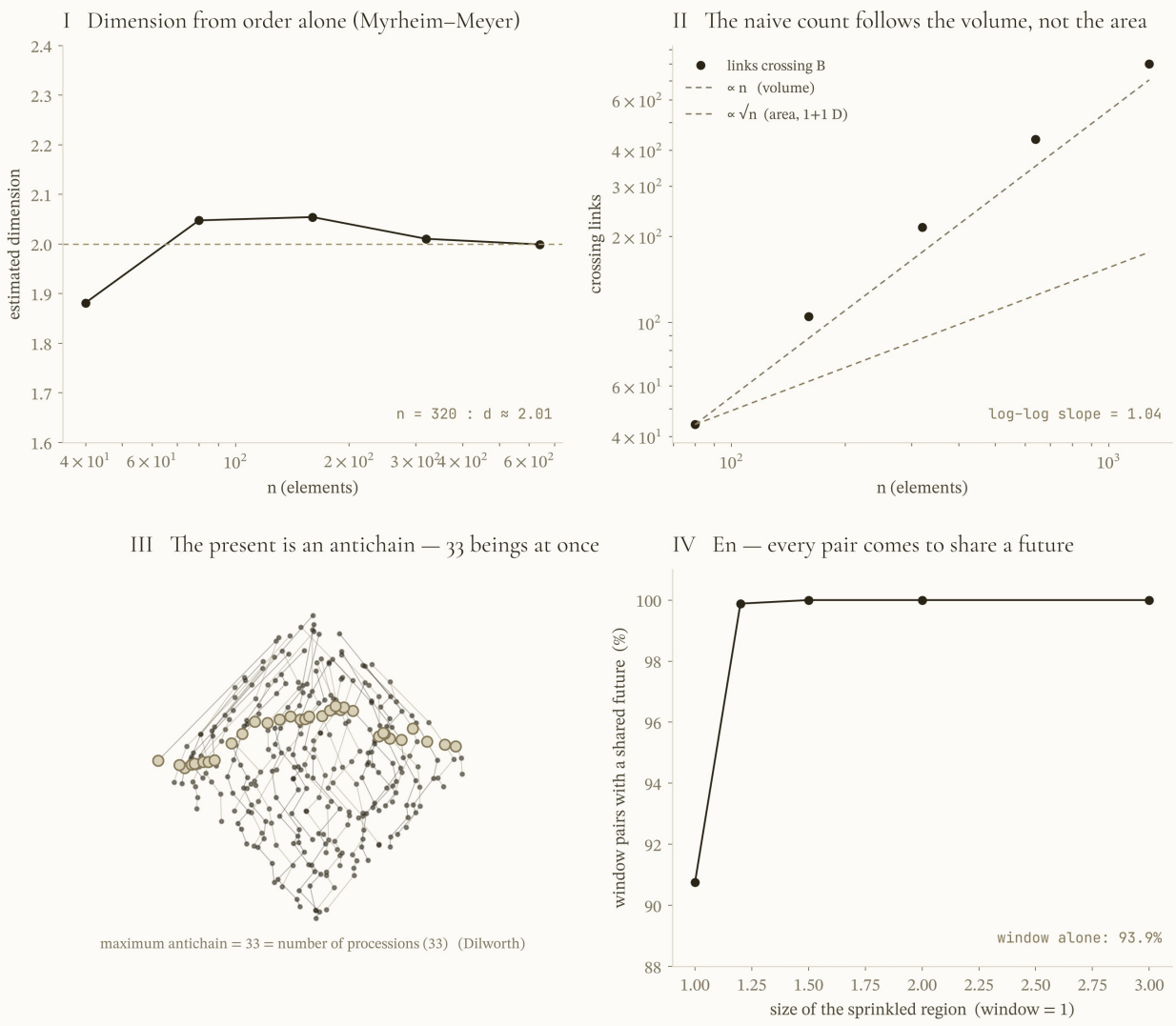

Fig. 12 Four extensions. I Dimension estimated from order alone (Myrheim–Meyer). II Crossing links scale with the volume (a negative test of the area law). III The maximum antichain = the present (gold, 33 elements). IV Fraction of window pairs sharing a future.

SECTION 9

Cultural Evolution and Diversity — A4 in Time

A4 stated the non-collapse of diversity for a structure at an instant. Stated for a time evolution, the same thing becomes the central problem of cultural evolution. The Moran process — one individual reproduces, one is replaced — homogenises inexorably without variation, and the entropy of types falls through the floor. With a mutation rate μ small but positive, the entropy keeps fluctuating above the floor (Fig. 9). For the procession to remain a column, new threads must keep being added. A4 is not a conservation law but a condition of supply.

CULTURAL EVOLUTION: PRESERVING DIVERSITY (A4 IN TIME)

```
def moran_entropy(n_types=8, pop=200, steps=400, mutation=0.02, seed=5):
    # Randomness
    rng = np.random.default_rng(seed)
    # Initialise the population evenly across n_types types
    pop_state = np.repeat(np.arange(n_types), pop // n_types)
    # Record entropy at each step
    H = []
    # Moran process: one individual reproduces, one is replaced
    for _ in range(steps):
        # Individual chosen to reproduce (random)
        parent = rng.integers(len(pop_state))
        # Individual chosen to be replaced (random)
        gone = rng.integers(len(pop_state))
        # With probability 'mutation' a new type appears (variation); otherwise inherit the parent's type
        pop_state[gone] = rng.integers(n_types) if rng.random() < mutation else pop_state[parent]
        # Type frequency distribution
        P = np.bincount(pop_state, minlength=n_types) / len(pop_state)
        # Record Shannon entropy (drifting to 0 means homogenisation = violation of A4)
        H.append(float(-(P[P > 0] * np.log(P[P > 0])).sum()))
    # Return the time series
    return np.array(H)
```

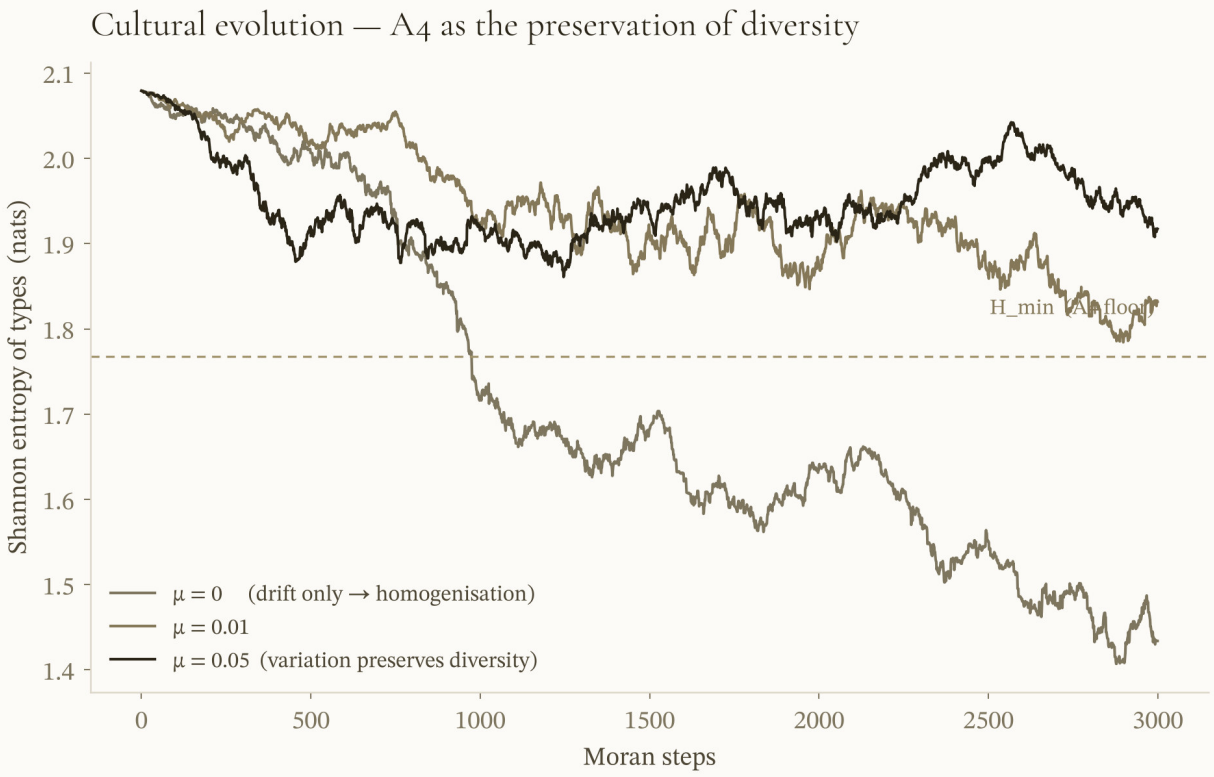

Fig. 9 Cultural evolution — entropy of types in a Moran process. Without variation the population homogenises (violating A4); a little variation preserves diversity.

SECTION 10

Testability and Limits

What this paper computed: the order, rank and links of a sprinkled causal set; the split by a null surface and the one-way crossing of threads; the implementation of f; the observables; checks of A1, A2, A4, A5 and the neighbourhood check of A3; the phase diagram; $\varkappa$_max by maximal flow; the T-duality formula; the order structure of the Tree of Life; the mandala projection; numerical confirmation of the window-edge theorem; $\tau = 0$; Dilworth's chain decomposition; the maximum antichain; the fraction of pairs sharing a future; the Myrheim–Meyer dimension; the n-dependence of crossing links; the entropy of a Moran process. All of it is reproducible from the attached code.

What it did not compute: (i) the derivation of the thresholds r_min, r_max, H_min, σ_max, which are at present empirical and should follow analytically from the Dou–Sorkin count; (ii) the uniqueness of f (steps (3) and (4) of §4.6); (iii) the extension from 1+1 to 3+1 dimensions — the horizon "area" becomes a true area only from 3+1, and as §5.4 shows the naive count does not reproduce the area law in 1+1; (iv) a formalisation of the correspondence with strings and branes; (v) nirvāṇa itself. The paper computes as far as it can and writes "not computed" where it cannot, so that the distance between artwork and specification is minimal. The only way to approach the truth is to say exactly what is known and where knowledge ends.

SECTION 11

Closing — What v10 Closes, and What It Opens

v9 wrote that a procession is a chain of causation inscribed in spacetime as the unique structure satisfying five axioms. v10 restates that sentence in computable form: a procession is the bundle of columns that, by Dilworth's decomposition, covers every element inside the region with a null surface in its past; the threads crossing the columns never exceed the horizon's area; and the first and the last of a column appear only at the edge of the observation window. Poetry and mathematics are saying the same thing — the intuition pursued since v1 lands in v10 on three facts of order theory: the window's edge, $\tau = 0$, and the chain decomposition.

Left open: the derivation of the thresholds, the proof of uniqueness, 3+1 dimensions, a formalisation of the string correspondence. And one thing outside computation — whether each of us in the column will treasure the threads between us. That is not an axiom; it is what the artwork keeps asking.

A procession is an order, inside a world with the horizon in its past, in which no one stands outside the column.

SECTION 12

Plate Gallery (v1–v10)

Each plate carries (i) the parameters (p, q, σ, κ), (ii) the observed change, (iii) the intent. The v1–v9 plates are redrawn with the drawing system of this edition (the v10 projection rule) from the parameters of each version, so that style, type and palette are uniform across all figures; the originals are kept under `Past/`. The diagrammatic plates v8–v9 are reconstructed in the same design.



Fig. 11 v10 plate — I the sprinkled causal set and the horizon, II the hologram H = f(S; B, θ_0), III pair correlation split by the horizon, IV the rank distribution. θ_0 = (0.5, 0.6, 0.01, 0.10), n = 320, seed = 7.

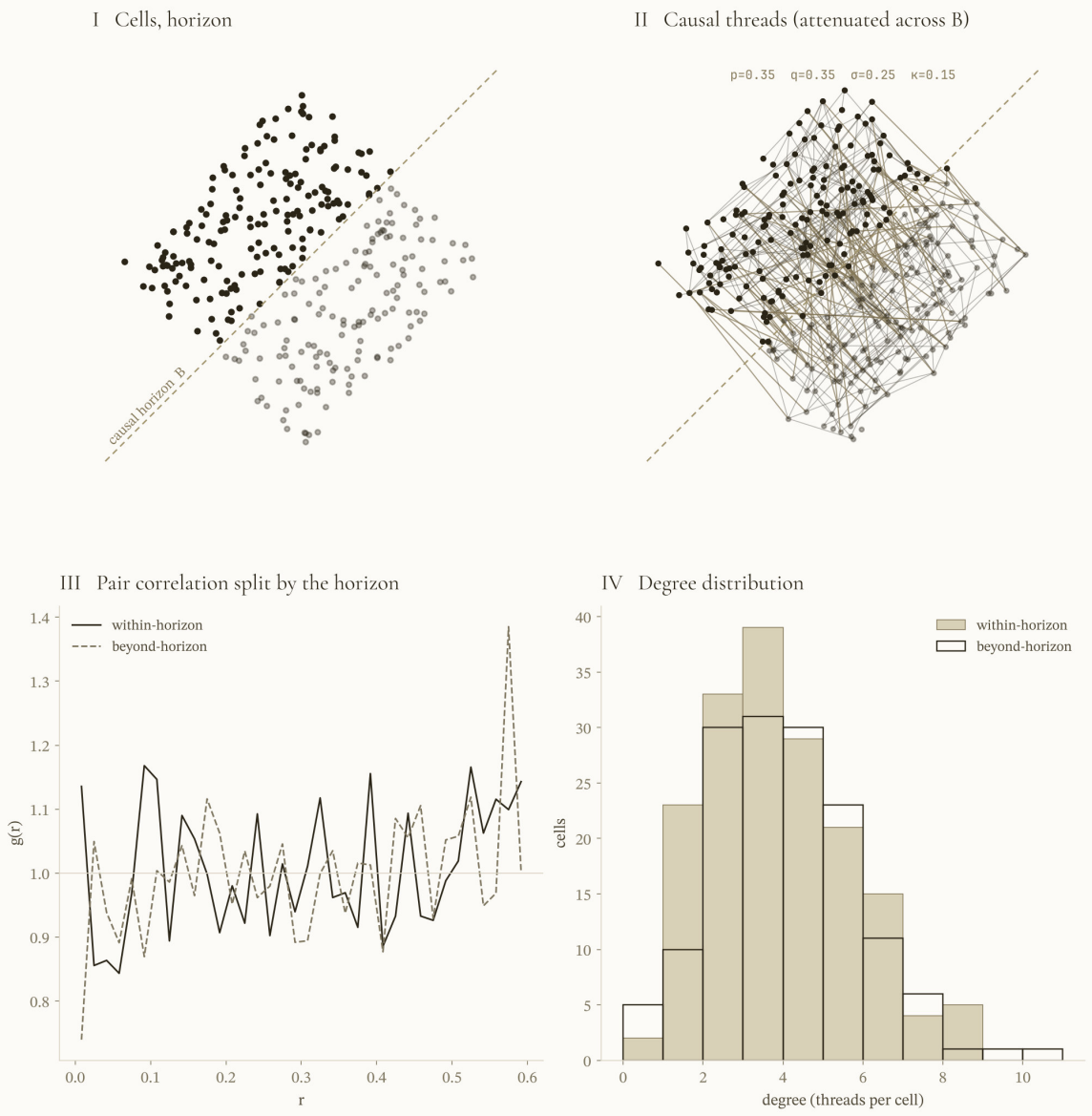

Plate v1 — baseline: two-phase split across the horizon

redrawn with the v10 projection rule from the v1 parameters · Dolice — Masaki Hirokawa

Fig. 13 Plate v1 (baseline) — two-phase split across the horizon

PARAMETERS	`p = 0.350  q = 0.350  σ = 0.250  κ = 0.150`
OBSERVED	A clear two-phase split across the horizon (structured / random).
INTENT	Fix a minimal baseline skeleton for comparison.

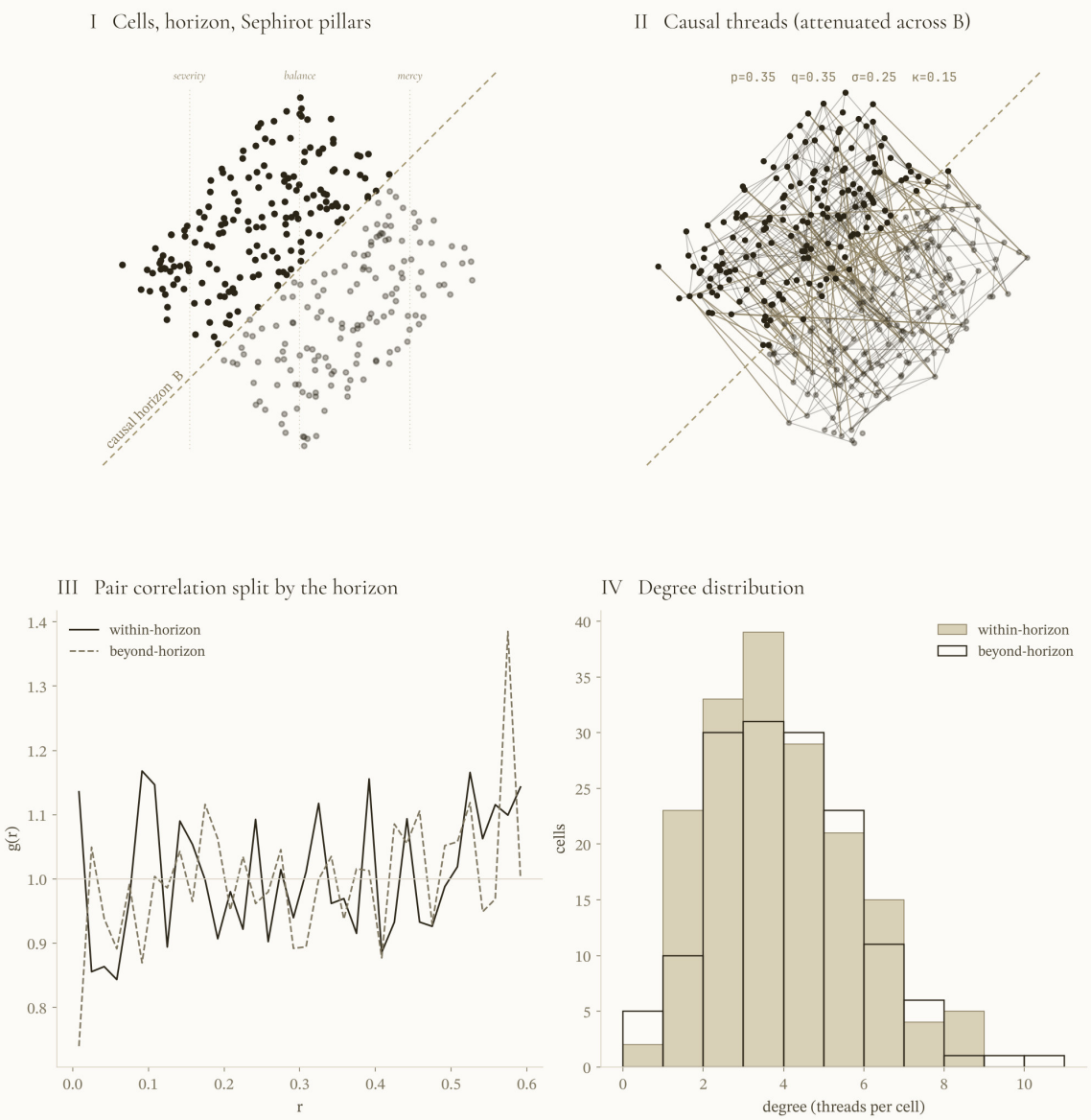

Plate v2 — Sephirot constraints: coherent density, increased order

redrawn with the v10 projection rule from the v2 parameters · Dolice — Masaki Hirokawa

Fig. 14 Plate v2 — coherent density and increased order under constraints

PARAMETERS	`p = 0.350  q = 0.350  σ = 0.250  κ = 0.150`
OBSERVED	Density becomes more coherent and order increases under constraints.
INTENT	Preserve visual causal readability via Sephirot constraints.

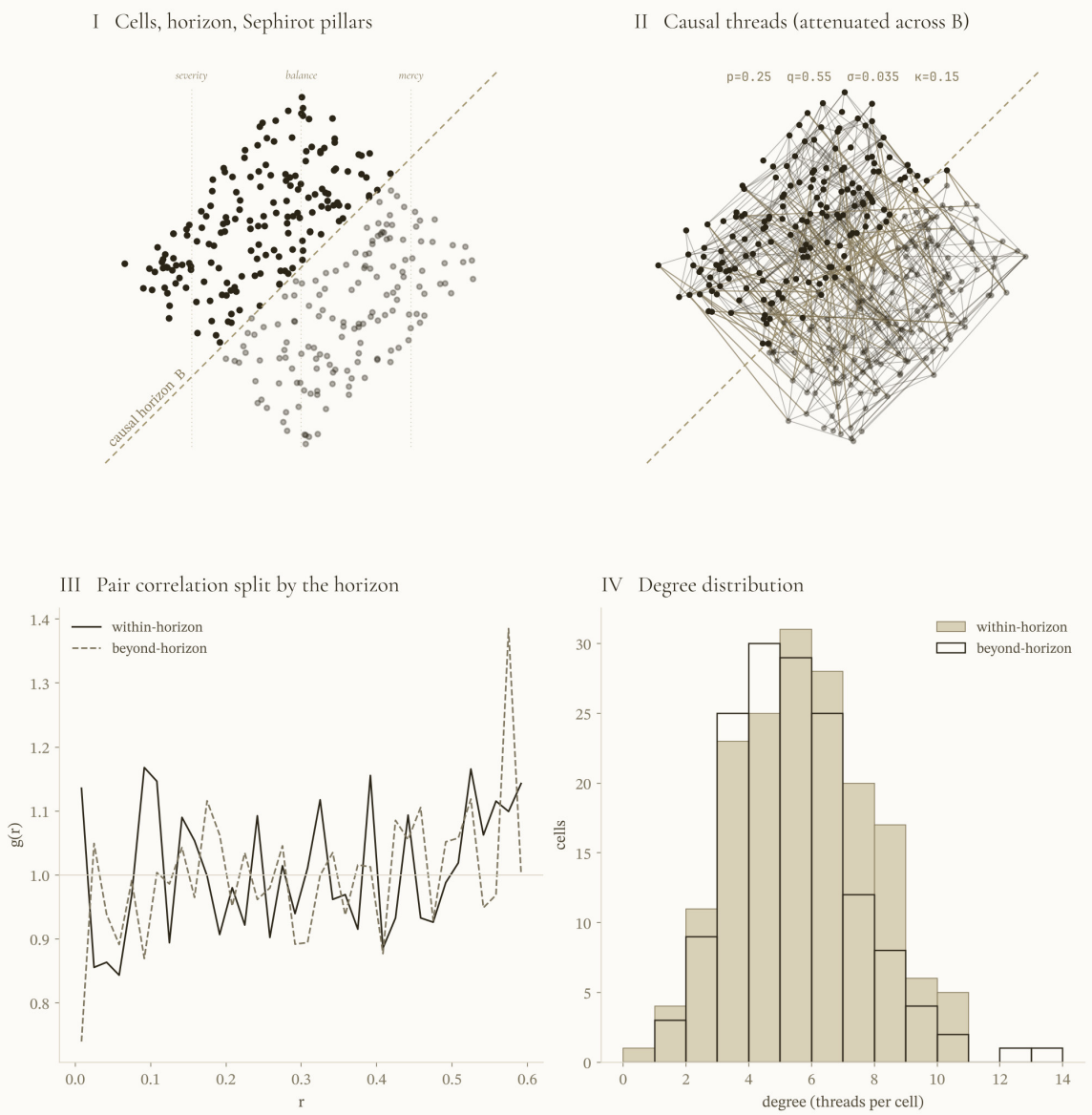

Fig. 15 Plate v3 — locality and haze in balance (p–q shift)

PARAMETERS	`p = 0.250  q = 0.550  σ = 0.035  κ = 0.150`
OBSERVED	A balance emerges between locality and haze.
INTENT	Let hardness and ambiguity coexist through p, q, and σ.

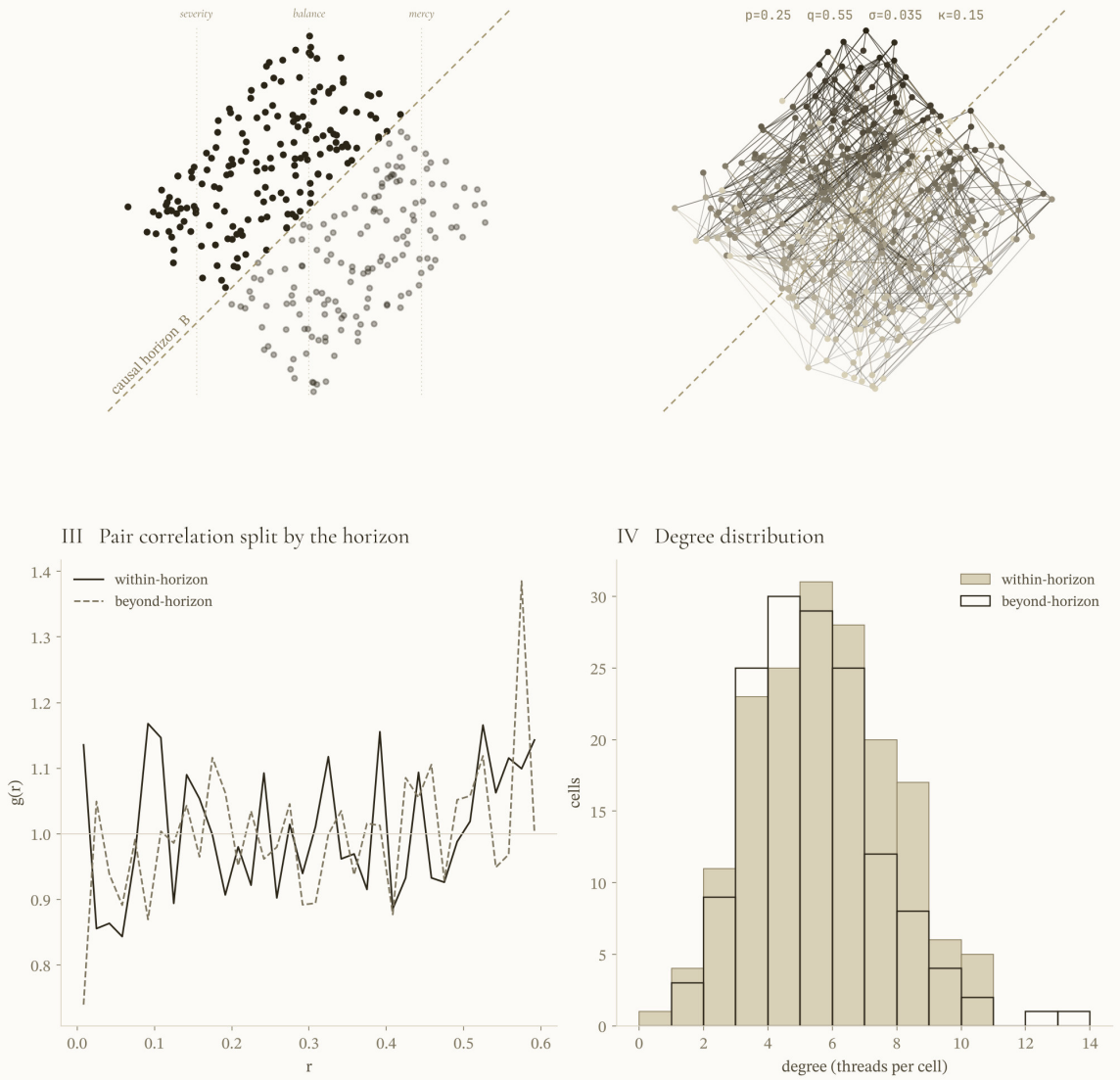

Fig. 16 Plate v4 — reachability becomes more legible

PARAMETERS	`p = 0.250 q = 0.550 σ = 0.035 κ = 0.150`
OBSERVED	Reachability becomes more legible.
INTENT	Reflect causal directionality in the visualization.

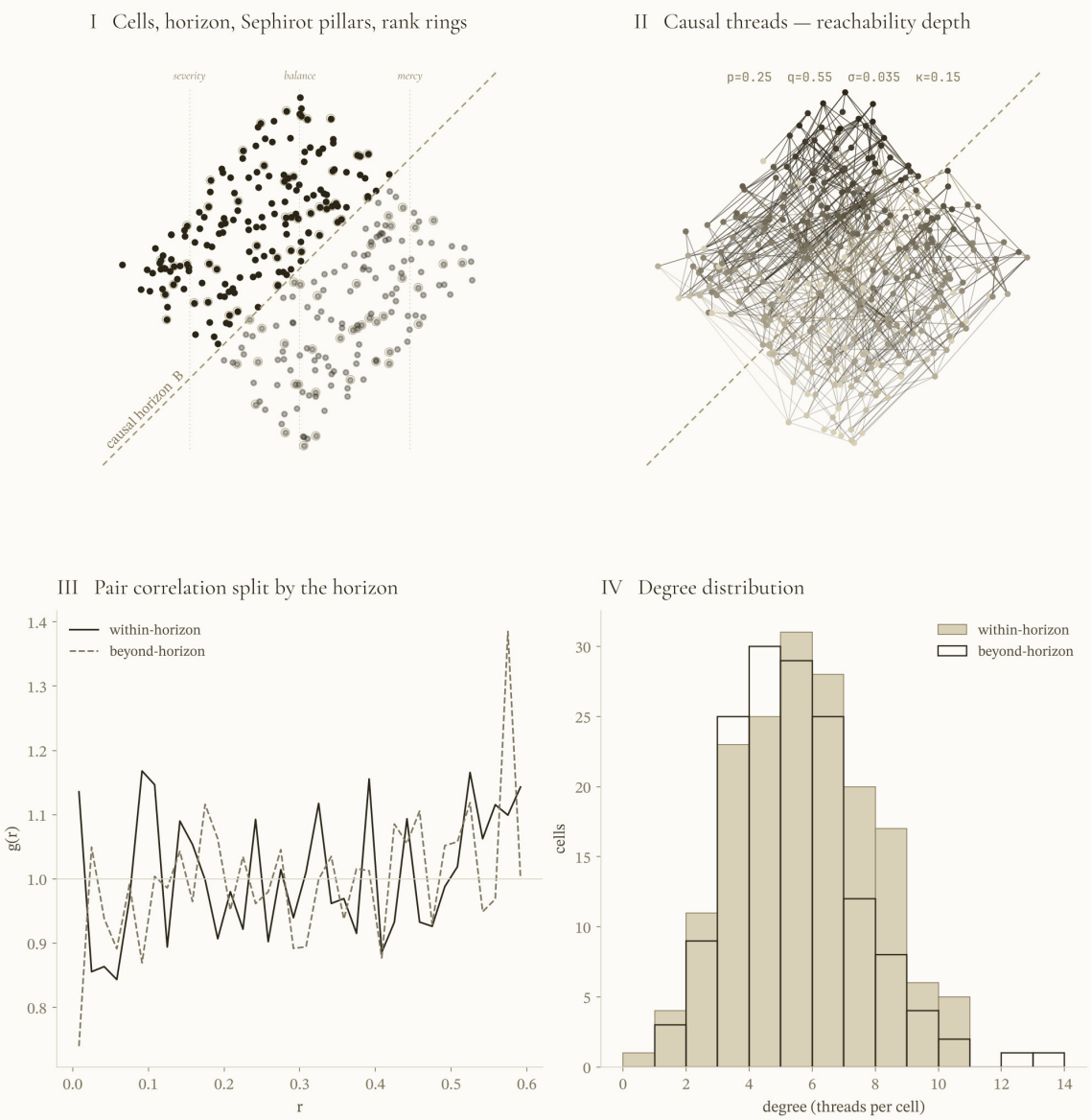

Fig. 17 Plate v5 — rank repetition becomes prominent

PARAMETERS	`p = 0.250  q = 0.550  σ = 0.035  κ = 0.150`
OBSERVED	Rank repetition becomes prominent.
INTENT	Organize the procession as a nested, repeating structure.

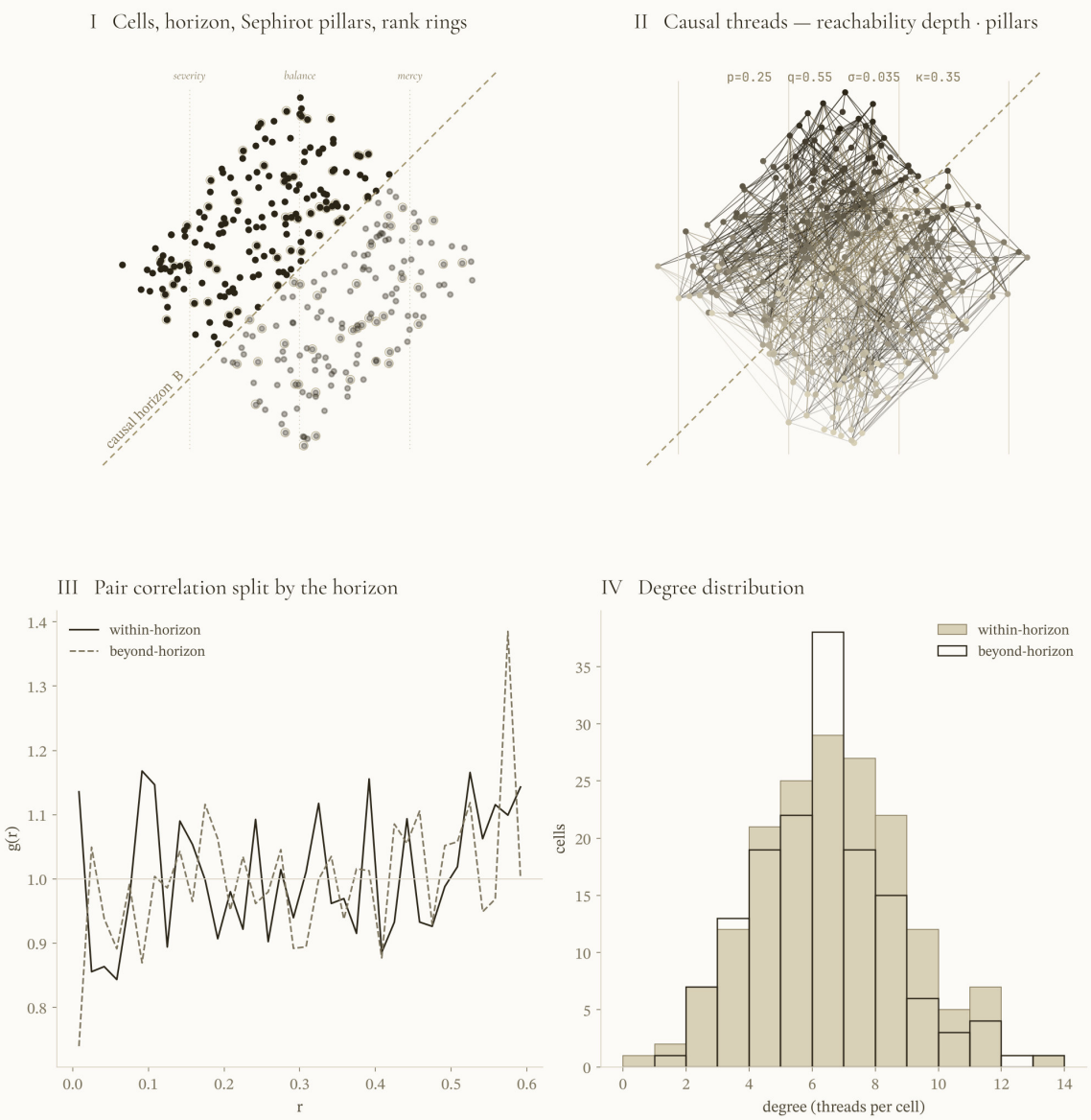

Fig. 18 Plate v6 — κ controls cross-horizon interference

PARAMETERS	`p = 0.250  q = 0.550  σ = 0.035  κ = 0.350`
OBSERVED	Cross-horizon interference becomes controllable.
INTENT	Treat tearing as an intentional, κ-mediated effect.

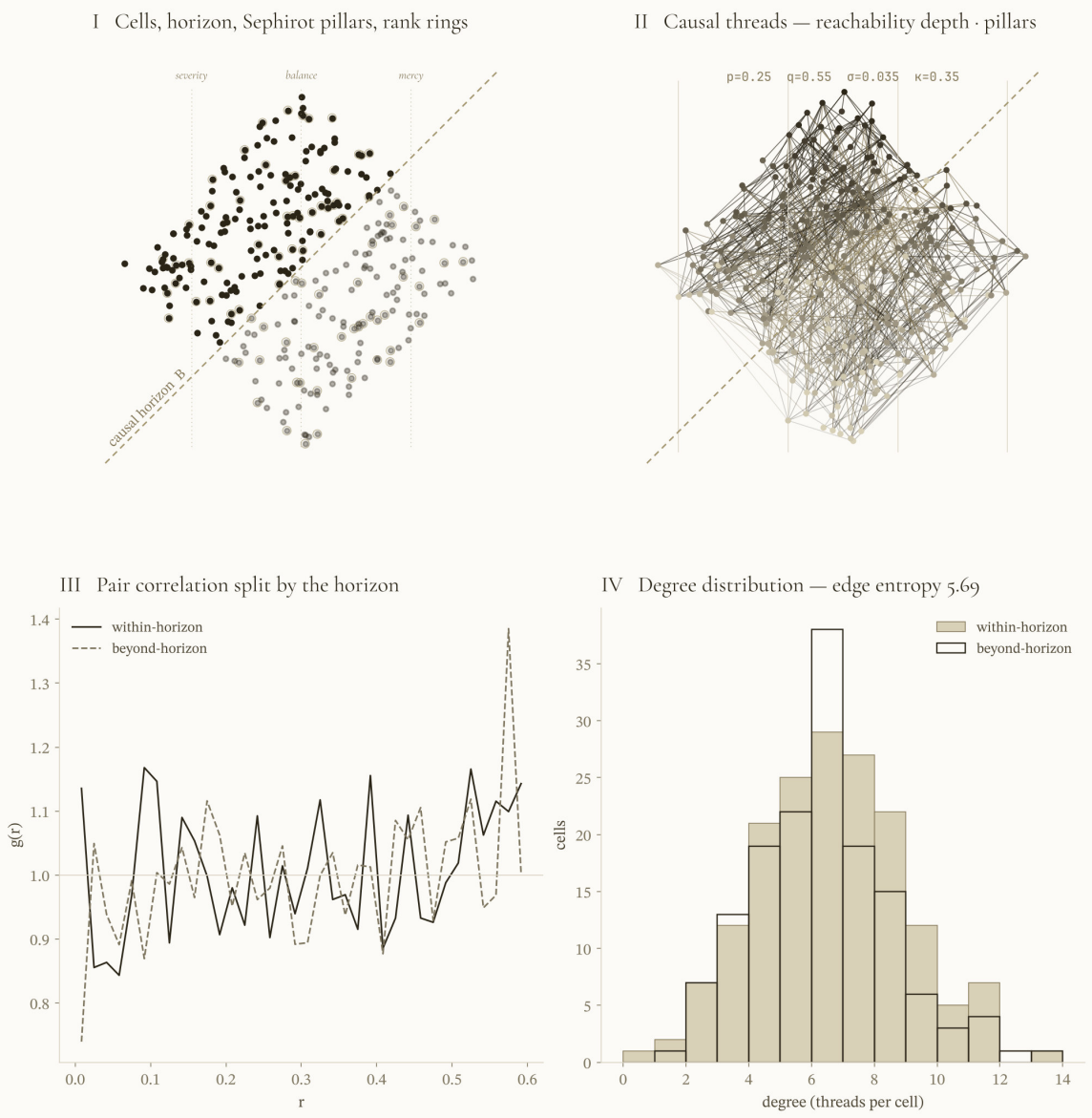

Fig. 19 Plate v7 — diversity maintained via edge entropy

PARAMETERS	`p = 0.250 q = 0.550 σ = 0.035 κ = 0.350`
OBSERVED	Monotony becomes less likely.
INTENT	Maintain diversity via edge entropy.

Plate v8 — Model → Output mapping clarified

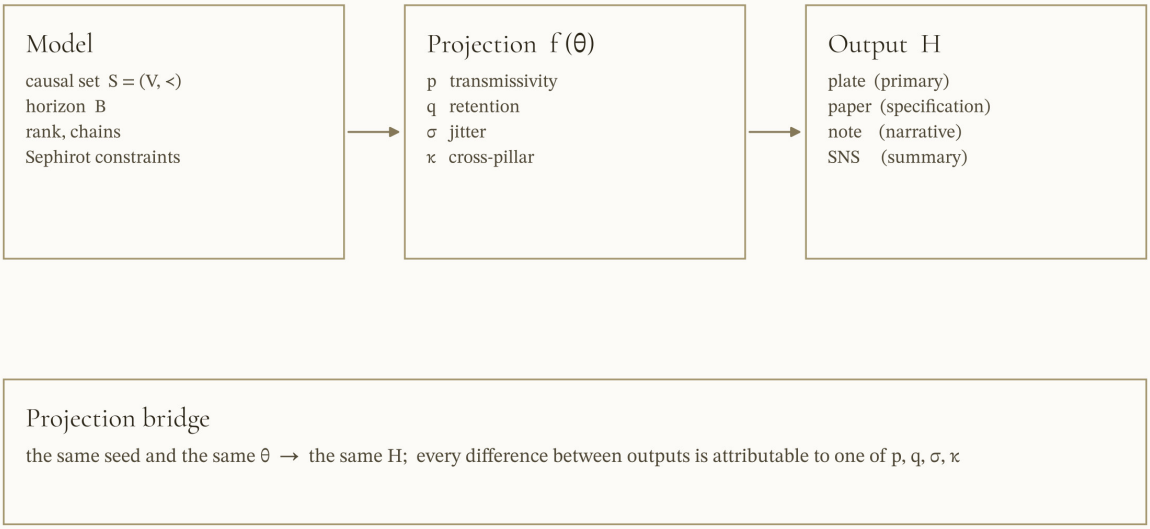

Fig. 20 Plate v8 — Model → Output mapping clarified

PARAMETERS	p = 0.500 q = 0.450 σ = 0.350 κ = 0.280
OBSERVED	The Model → Output correspondence becomes clearer.
INTENT	Make the projection bridge explicit.

Plate v8.1 — MVC remapping of the v7 specification

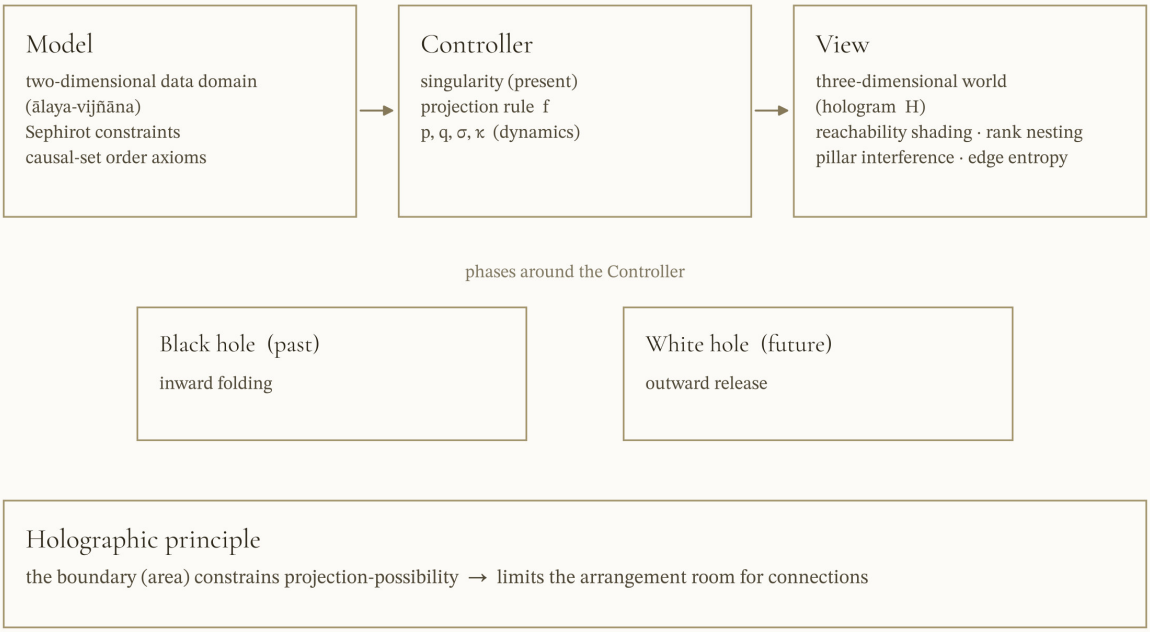

Fig. 21 Plate v8.1 — interpretation stabilizes; MVC correspondence fixed

PARAMETERS	`p = 0.500  q = 0.450  σ = 0.350  κ = 0.280`
OBSERVED	Interpretation stabilizes.
INTENT	Recover v7 and fix the MVC correspondence.

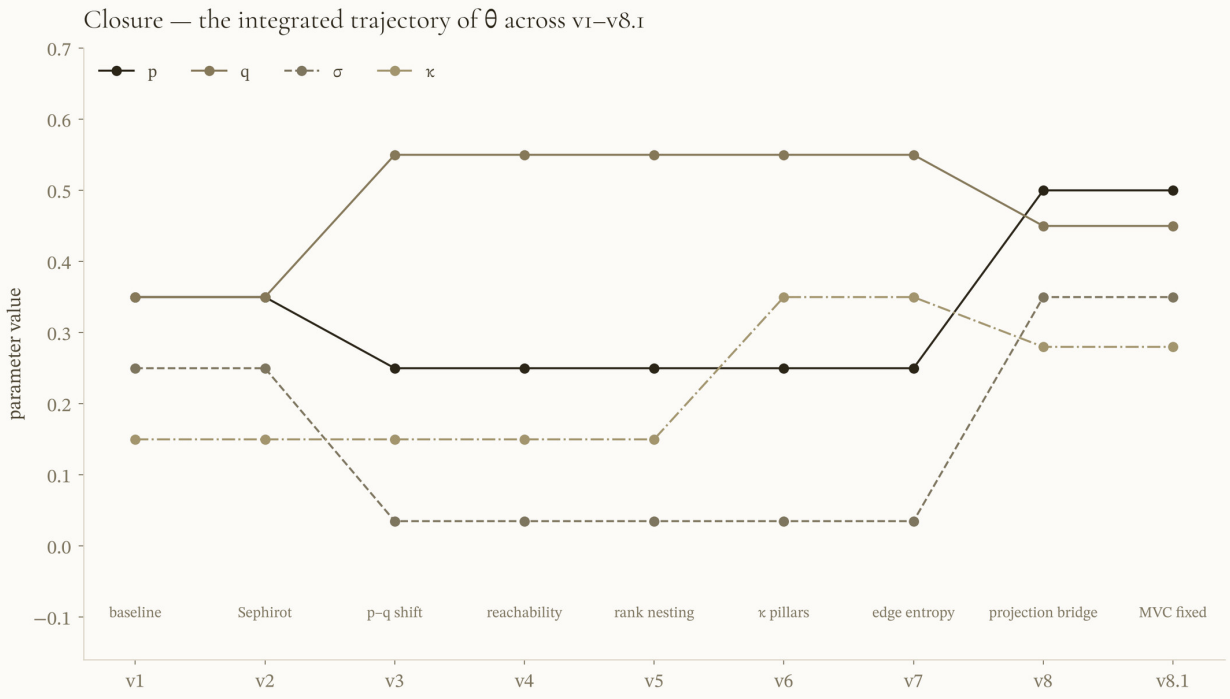

Fig. 22 Closure — integrated trajectory (v1–v8.1)

PARAMETERS	`p = 0.500  q = 0.450  σ = 0.350  κ = 0.280`
OBSERVED	The overall trajectory is presented in an integrated view.
INTENT	Conclude with a panoramic overview of v1–v8.1.

Plate v9 — projection rule f, stability domain, counterexamples

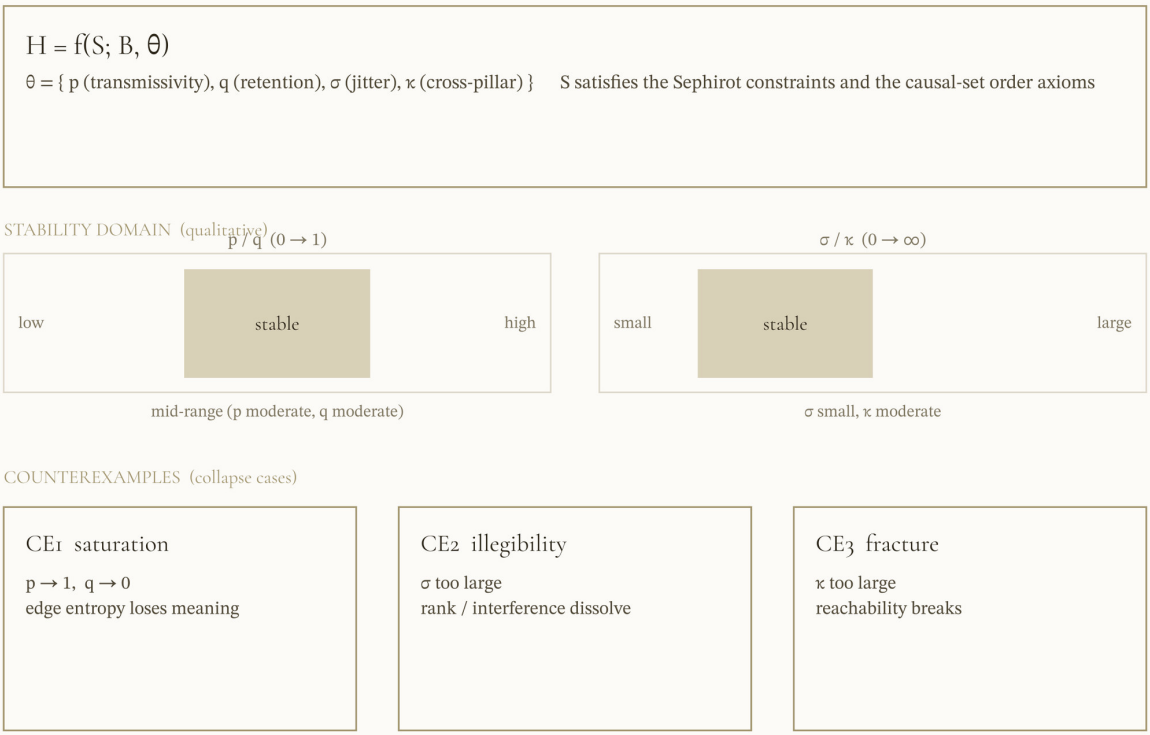

Fig. 23 Plate v9 — projection rule f, stability domain, counterexamples

PARAMETERS	`p = 0.500  q = 0.450  σ = 0.350  κ = 0.280`
OBSERVED	Axioms, stable regime and counterexamples fit on one plate.
INTENT	Reconstruct f from five axioms and close the series.

Appendix A — Running the Code

The code of this paper is collected in a single file, chfp.py (chfp_en.py with English annotations). The only dependencies are numpy, networkx and matplotlib. figures.py calls chfp.py to generate the figures. With the seeds unchanged, every number and figure in this paper is reproduced as is.

```
RUN
pip install numpy networkx matplotlib
python3 figures.py                      # every figure and plate → Figures/
python3 -c "import chfp as m; print(m.window_test())"
```

Stage	Functions	Section
Model	`sprinkle_diamond · causal_matrix · links · rank_longest_chain`	§1
Dimension	`ordering_fraction · myrheim_meyer_dimension`	§1.2
Horizon	`horizon_split · cross_links · horizon_area`	§2.1
Projection f	`Controller · project · observe`	§2.2–2.3
Observables	`edge_entropy · reachability_depth · pair_correlation`	§2.3
Axioms	`check_axioms · is_interior · is_stable · phase_diagram`	§4.5
Bound	`max_threads_across · kappa_max · area_law`	§5.2, §5.4
Strings	`closed_string_spectrum · t_duality_holds`	§6.2
Mandala, Tree	`tree_of_life · mandala_layout`	§7
Unborn, unending	`window_test · null_proper_time`	§8.1–8.2
Procession, present, en	`minimum_chain_cover · chains_from_matching · maximum_antichain · common_future_fraction · en_window_test`	§8.4–8.6
Cultural evolution	`moran_entropy`	§9

Appendix B — Glossary

因果集合 / causal set	A set of elements with a partial order $\prec$; a model of discrete spacetime in which order and number alone fix the geometry.
散布 / sprinkling	Dropping points by a Poisson process in number proportional to spacetime volume, with the order fixed by light cones.
リンク / link	A causal relation with no intermediary; an edge of the Hasse diagram.
rank	The length of the longest chain into the past; the tier of the procession.
地平面 B / horizon	The null surface $u = u_h$; threads cross only from outside to inside.
$\theta = (p, q, \sigma, \kappa)$	Transmissivity, retention, jitter, interference: the Controller of the projection rule f.
Ω_stable	The set of θ on which A1–A5 hold at once.
ビット・スレッド / bit threads	The maximal number of independent threads across a boundary = the minimum cut (Freedman–Headrick).
Dou–Sorkin の数え方	The proposal to count black-hole entropy by the number of links crossing the horizon.
Dilworth の定理	A finite partial order can be partitioned into as many chains as its width.
T 双対 / T-duality	The string at radius R and at α'/R have the same spectrum.
反鎖 / antichain	A set of mutually unrelated elements. The maximum antichain's size = the width = the number of processions (Dilworth); in this paper, the definition of the present.
Myrheim–Meyer 次元	Estimating the dimension d by inverting the fraction of related pairs, $f(d) = \Gamma(d+1)\Gamma(d/2)/(2\Gamma(3d/2))$.
縁（本稿）/ en	The existence of a shared future for two elements; $C\,C^{T} > 0$.
窓の縁の定理（本稿）	The fraction of minimal and maximal elements tends to zero as the region is enlarged with the window fixed.

References

L. Bombelli, J. Lee, D. Meyer, R. D. Sorkin, “Space-time as a causal set,” Phys. Rev. Lett. 59 (1987) 521.

R. D. Sorkin, “Causal sets: Discrete gravity,” in Lectures on Quantum Gravity (2005); arXiv:gr-qc/0309009.

D. Dou, R. D. Sorkin, “Black hole entropy as causal links,” Found. Phys. 33 (2003) 279; arXiv:gr-qc/0302009.

J. D. Bekenstein, “Black holes and entropy,” Phys. Rev. D 7 (1973) 2333. S. W. Hawking, “Particle creation by black holes,” Commun. Math. Phys. 43 (1975) 199.

S. Ryu, T. Takayanagi, “Holographic derivation of entanglement entropy from AdS/CFT,” Phys. Rev. Lett. 96 (2006) 181602.

M. Freedman, M. Headrick, “Bit threads and holographic entanglement,” Commun. Math. Phys. 352 (2017) 407; arXiv:1604.00354.

J. Maldacena, “The large N limit of superconformal field theories and supergravity,” Adv. Theor. Math. Phys. 2 (1998) 231.

E. Witten, “String theory dynamics in various dimensions,” Nucl. Phys. B 443 (1995) 85.

J. Polchinski, String Theory, Vols. I–II, Cambridge University Press (1998). （T 双対・D ブレーン）

R. P. Dilworth, “A decomposition theorem for partially ordered sets,” Ann. Math. 51 (1950) 161. D. R. Fulkerson, “Note on Dilworth's decomposition theorem,” Proc. AMS 7 (1956) 701.

J. Myrheim, “Statistical geometry,” CERN preprint TH-2538 (1978). D. A. Meyer, The Dimension of Causal Sets, PhD thesis, MIT (1988).

D. Kőnig, “Gráfok és mátrixok,” Matematikai és Fizikai Lapok 38 (1931) 116.

P. A. M. Moran, “Random processes in genetics,” Proc. Camb. Phil. Soc. 54 (1958) 60.

『般若波羅蜜多心経』（玄奘訳）「不生不滅 不垢不浄 不増不減」。

M. Hirokawa, Causal Horizon Funeral Procession — Complete Edition (v9), Dolice (2026). https://dolice.net/pub/

Dolice

Causal Horizon Funeral Procession — Complete Edition v10

Masaki Hirokawa（廣川政樹）

All figures and numbers regenerated from the attached code. All artworks are created entirely by hand, without the use of AI.

https://dolice.net · 2026