

Interaction Nets That Carry Their Own Coherence

A Design-and-Test Program for n-Cell Confluence over Cubical HoTT and the Boundary Polygraph

Institute for Applied Ontological Mathematics

2026 **[living document — updated as the program proceeds]**

Abstract

Standard interaction nets are tested for the wrong thing. Confluence — the guarantee that the order of reduction does not change the answer — is almost always checked as a *diamond* (Church–Rosser): two one-step reductions of the same redex reconverge. But our nets are not standard. Boundary is a *typed, open* reduction calculus that compiles kernel-checked Lean proofs into interaction nets, and its confluence already lives one categorical level higher: each critical pair is a *classifier square*, each resolution is an *admissible Squier join* that carries its own legality receipt, and the whole convergent system *is* an idempotent free-compute closure. This document does three things. It *reviews* what makes our nets unique and why a diamond test under-measures them. It proposes a *design and test methodology* that checks confluence at its true level, by moving the critical-pair squares into Cubical Homotopy Type Theory — where the Kan filler hcomp *is* the admissible join, and the h-level tower *is* the confluence-coherence tower. Triaging the multiway-rewriting literature then sharpens the destination: confluence is not a ladder but a *plane* — a vertical coherence tower (where forks join: WCR[1] through n-cell, the algebraic, topological, geometric, and logical coherences no rewriting system has had the instruments to reach) crossed with a horizontal multiway/branchial axis (where, under genericity, side effects, or measurement, forks deliberately do not), the two meeting at the quantum. The plan is staged and each stage is tested to completion; this is the map we update as we build.

1 The question: what is the right test for our nets?

Question. An interaction net computes by firing *active pairs* — two agents meeting at their principal ports — until none remain. The deep property that makes this trustworthy is *confluence*: fire the pairs in any order, get the same normal form. For ordinary nets that property is a *diamond*: a one-step fork reconverges in one step (Figure 1). The diamond is the whole test, and it is enough — for ordinary nets.

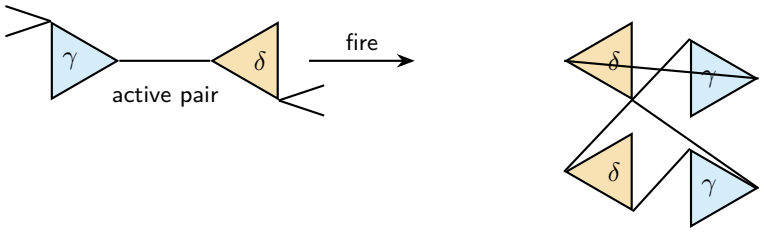


Figure 1: **Act 1 — the picture with zero prerequisites.** An interaction net computes by firing *active pairs*: two agents touching at their principal ports (left). One firing of a γ–δ pair *commutes* them into four agents (right); annihilation of like agents instead erases the pair. Confluence asks: does the final net depend on the firing order? For ordinary nets the answer is secured by a one-step diamond. Our nets need more.

Claim. Our nets are not ordinary, and the diamond under-measures them. Boundary carries *types* along each reduction and admits *open* nets with free ports; it is the substrate that runs Lean proof-terms as nets. We have already shown (in a companion development on admissible Squier certificates) that its confluence lives at the level of *parallel* 3-*cells agreeing*, not 2-cell reconvergence; that each critical pair is a *classifier square*; and that the convergent system is, underneath, an idempotent closure you can run for free a second time. A diamond check confirms none of this. (The baseline is, in fact, already stronger than generic Church–Rosser: Lafont's three conditions — linearity, binary interaction, and no ambiguity — give *uniform* confluence WCR^1, a fork that reconverges in *exactly one* step each side. The question is therefore not whether forks join, but how far *above* WCR^1 the coherence runs — and, as Section 4 shows, how far *sideways* the branching spreads when uniform confluence is deliberately broken.) The right instrument tests at the level the structure actually occupies — and we already own that instrument, because the *h-level tower* of Cubical Homotopy Type Theory is exactly the confluence-coherence tower.

Meaning. If we test confluence at its true level, three things follow that a diamond cannot give. We can *certify* that a resolution is legal, not merely that a common reduct exists. We can *grade* how deep a net's coherence runs — a single number that tells a scheduler how aggressively it may reorder work. And we can *extend* the ladder: 2-cell, 3-cell, ..., n-cell confluence, where the higher rungs are the algebraic, topological, geometric, and logical coherences (Figure 6). This is territory no rewriting platform has fully entered, because none has carried its coherence at the level where these rungs live. Ours does.

2 What our nets are, exactly

A typed, open calculus that runs proofs. Four features separate Boundary from a textbook interaction net.

1. **Type-decorated histories.** A reduction $\mathrm{net}_0 \to \mathrm{net}_1 \to \cdots$ carries type information at every step; the net is not a bare graph but a graph over a type.
2. **Open nets.** Free ports are first-class: a net may be *incomplete*, with dangling interfaces (the "open horns" of the homological reading). Computation proceeds up to the open boundary and docks when the missing data arrives.

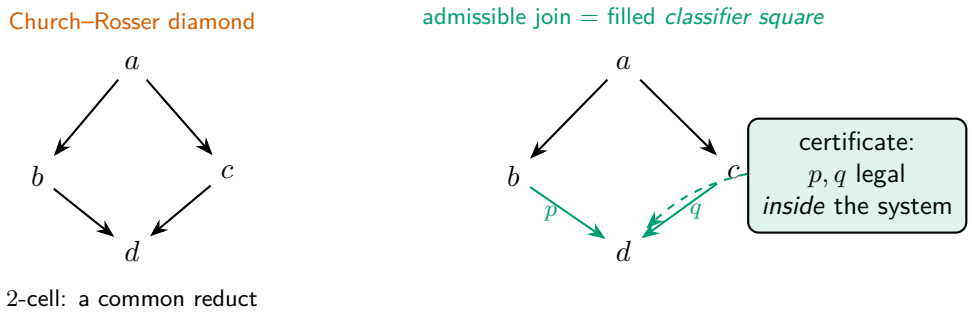


Figure 2: **Act 2 — the structure that does the work.** Left: the textbook diamond closes a fork with a common reduct — a 2-cell, no receipt. Right: our *admissible join* closes it with certified closing paths p, q *and* a proof they are legal moves of the system. The fork is a *classifier square*; its filler is the certificate. Confluence becomes "every square has a legal filler."

3. **It compiles proofs.** Boundary translates Lean dependent-type-theory terms — kernel-checked proofs, which are programs — into interaction-net form. The net is the executable shadow of a verified theorem.
4. **Gray-categorical confluence.** Its critical-pair theory is the finite Forest–Mimram theory of rewriting in a Gray category, not the first-order diamond.

The fourth point is the one that changes the test, so it gets its own section.

3 Confluence beyond the diamond

The diamond is a 2-cell; ours is a 3-cell. A Church–Rosser diamond closes a local fork $b \leftarrow a \rightarrow c$ with a common reduct d (Figure 2, left). That is a 2-cell. But "the order of work does not matter" is really a statement about *which closing path you take*, and that is a 3-cell: two different ways of resolving the same ambiguity must themselves agree. Our companion result states confluence at exactly this level.

Theorem 3.1 (Admissible Squier coherence — four-layer)**.** (Picture.) *A net's ambiguities are harmless not because forks reconverge, but because every fork's resolution carries a* receipt *that its closing paths are legal moves of the system (Figure 2, right).* (Statement.) *If the system terminates and every local peak has an* admissible join *— an apex d with closing paths and admissibility certificates — then it is confluent; equivalently, parallel 3-cells over the presentation agree.* (Meaning.) *This is Squier's theorem with the vacuous "fullness" hypothesis deleted: coherence is carried locally, by the fillers, with a certificate, rather than asserted globally.* (Proof.) *Machine-checked in Lean over the Forest–Mimram substrate; the closing paths' legality is the only place fullness was ever spent, and the admissibility certificate supplies it exactly.*

And underneath, an idempotent closure. The same companion result shows the punchline: every convergent system *is* a normalization *nucleus* — an idempotent closure R with $R \circ R = R$, "a closure you can run for free a second time." This is the free-compute principle (the $\varepsilon = 2e - 1$ identity that turns an involution into an idempotent), and it is the cheapest possible confluence test: not "do the forks join," but "is normalization idempotent."

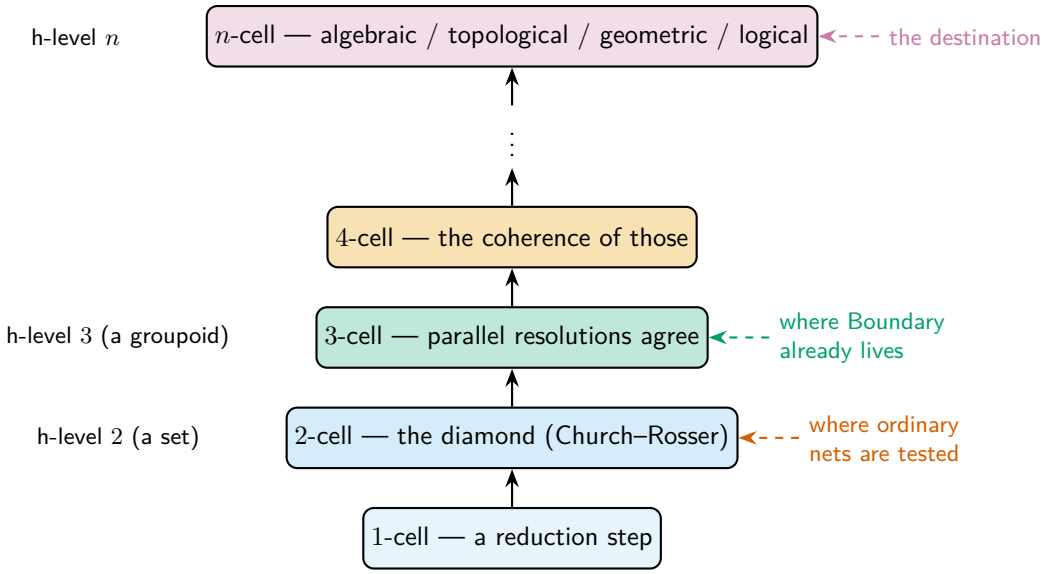


Figure 3: **The confluence tower is the h-level tower.** Each rung is a coherence of the rung below. The diamond is rung 2; our admissible-join coherence is rung 3. The *height* a net reaches is its coherence depth, and Cubical HoTT measures it: this is literally the ViewData-hlevel law from the formal-philosophy development, reused as a confluence instrument.

The coherence tower. Stack the levels (Figure 3). A 1-cell is a reduction; a 2-cell is a diamond; a 3-cell is "parallel resolutions agree"; an n-cell is the coherence of the coherences. The height a given net reaches is its *coherence depth*. We *conjecture* this is the same tower as the *h-level* tower of Cubical HoTT: a set is 2-coherent, a groupoid 3-coherent, and so on. The h-level instrument itself we own — it was built and machine-checked in the formal-philosophy development, where a *view-data* coherence tower's height was shown to equal a carrier's homotopy depth. What is *not* yet a theorem — and is the first obligation of this program — is that a Boundary net's *confluence* coherence is faithfully read by that instrument (no Boundary system has yet been pinned to an h-level). Until that bridge is proved or refuted, the coherence depth is a conjecture, not a measurement; the build's first component is to settle it, and nothing downstream may treat a depth as genuine before it does.

4 The second axis: where forks do not join

Confluence is not a ladder but a plane. The tower of Section 3 answers "when forks *join*, how deeply do they cohere?" But the richest nets have a second, orthogonal freedom: forks that *do not* join, by design. Three mechanisms break uniform confluence on purpose, and each turns the failure into content. *Genericity*: a generic agent — an arbitrary agent standing for a function variable — buys higher-order rules and currying, at the cost of single-step reconvergence (Jiresch and Gramlich, 2011). *Side effects*: an impure feature is a monad, and a monad's sequencing is a non-confluent commitment. *Measurement*: the pinch that collapses quantum coherence is exactly a fork whose branches are real outcomes, not artifacts. In each case the object that records the branching is a *multiway system* (Gorard et al., 2020).

The multiway/branchial complement. A multiway evolution graph keeps *every* reduction path (states merged by isomorphism); its *branchial* graph — the quotient by common ancestry —

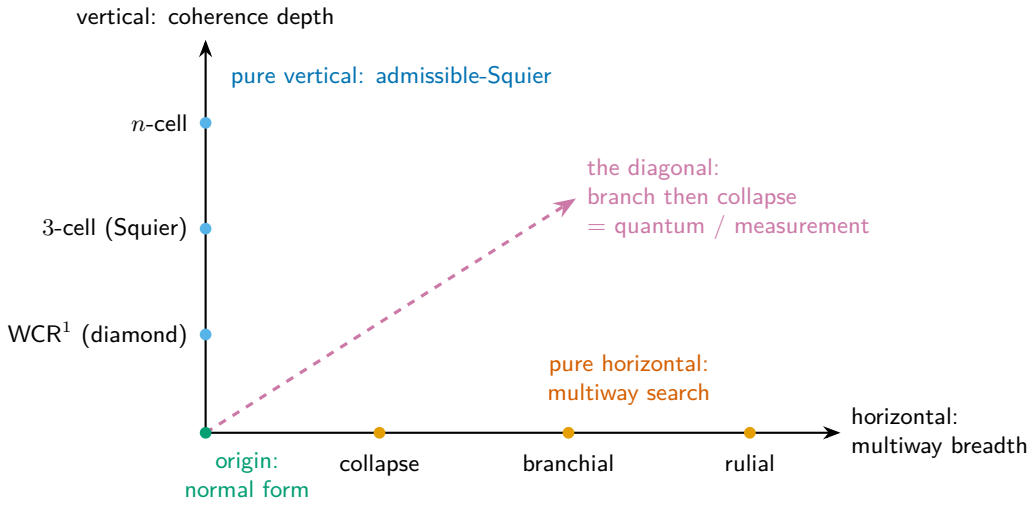


Figure 4: **Confluence is a plane.** The vertical axis (Section 3) grades how deeply forks cohere when they *join* (WCR$^1 \to$ 3-cell $\to$ n-cell); the horizontal axis grades how far they *branch* when they do not (collapse $\to$ branchial $\to$ rulial). The diagonal — a branch followed by a collapse — is the quantum: superposition (branchial) read out by measurement (the vertical pinch). Our quantum substrate lives on the diagonal; the admissible-Squier coherence on the vertical; open-ended multiway search on the horizontal.

is the quantum-like superposition of histories. The causal relationships between productions (p_2 consuming a (hyper)edge that p_1 produced) form 2-cells, yielding a weak 2-category **MuCauGraph** with two monoidal structures: $\otimes_M$, parallel composition of productions, and $\otimes_C$, a *partial* monoidal product defined exactly on the *causally independent* pairs (Gorard et al., 2021). That partiality is not a defect — it *is* the scheduler's reorder budget: two steps may run in parallel precisely when $\otimes_C$ is defined on them.

The plane, and where the axes meet. Put the two together (Figure 4). The *vertical* axis is the coherence tower — reduce within fixed rules, forks join, graded WCR$^1 \to$ 3-cell $\to$ n-cell. The *horizontal* axis is the multiway — branch or transform the rules, forks spread, graded collapse $\to$ branchial $\to$ rulial. The origin is the confluent normal form; pure vertical is our admissible-Squier coherence; pure horizontal is open-ended multiway search; and the *diagonal* is the quantum, where a branch (horizontal) is read out by a measurement-collapse (the vertical pinch). This is not a metaphor we import. Gluing classifier squares yields a non-distributive (quantum) logic *without* Hilbert space (Gunji et al., 2026); the vertical/horizontal split is exactly inference-within-a-model versus transform-the-model; and an independent four-state cyclic algebra — cyclic order 4, with a retention-not-selection "wait" state for held contradiction — reproduces our re-entry $\sqrt{-1} = \langle i \rangle$ from a neuromorphic direction (Fellouri and Adjailia, 2026). The same object keeps arriving.

Instruments for the horizontal axis. The cubical bridges of Section 5 grade the vertical axis; the horizontal axis has its own catalogued instruments. *Generic typed rules* add higher-order matching under a port-type discipline with explicit *confluence-preserving constraints* (Jiresch and Gramlich, 2011) — a concrete test: does adding this generic rule preserve WCR^1, or open a branch? *Causal lemma selection* orders rewrites by their outgoing causal-edge count in the multiway graph, a measured heuristic that shortens downstream proofs (Gorard et al., 2021). And *token-*

Our object	Cubical HoTT native	What it buys
`BoundaryCriticalPair` (classifier square)	`Square` (a 2-cell, primitive)	critical pairs become checkable cells
Admissible join (certified filler)	`hcomp` (the Kan filler)	"joins inside the system" = box has a
Parallel 3-cells agree (depth)	the h-level tower (`ViewData-hlevel`)	*grade* how far beyond the diamond
The coherence nucleus (idempotent closure)	`pinch`-idempotency, `isProp`-collapse	confluence ⇔ a *cheap* idempotency te
Type-decorated open net	a fibration; reduction = `transport`	open horns = missing fibers; dock =

Table 1: **The bridges.** Each row turns a Boundary confluence object into a native cubical construct. The deepest is the third: the confluence-coherence tower and the homotopy h-level tower are the same ladder, so the instrument that grades one grades the other.

passing reduction reaches normal form *independent of strategy*, with read-back embedded in the net itself (Salikhmetov, 2016) — the operational face of confluence. These become the horizontal-axis gates of the test battery.

5 The four bridges to Cubical HoTT

Why move the test into Cubical HoTT rather than test in place? Because the structure we need is *native* there, not added (Table 1). Path composition is defined by the Kan operation hcomp; hcomp *is* horn-filling; and a filled horn *is* an admissible join. The instrument and the object are the same shape.

6 The design-and-test methodology

Design: model confluence at its level. Represent each critical pair as a cubical `Square` whose `hcomp` filler is its admissible join; an *illegal* resolution is then a square *with no filler*, caught by the type checker rather than by a missing side condition. Grade each net's coherence depth with the h-level instrument. Model the typed open net as a fibration whose reduction is transport, so that docking two torn shards is a Kan lift.

Test: a battery beyond the diamond. We never declare a phase done on a diamond alone. Five gates, in increasing depth (Figure 5):

1. **Nucleus / idempotency** (cheap, primary): normalization satisfies $R \circ R = R$. Run against the catalogued free-compute closure rows.
2. **Admissibility**: each join carries a legal in-system certificate — catching resolutions a diamond blesses but Squier-with-fullness would not.
3. **Coherence depth**: the h-level rung is measured and *must not regress*; a drop is a confluence regression even when the diamond still closes.
4. **Classifier-square**: each critical pair's classifier value agrees across the square — a decidable, topos-side check.
5. **Faithfulness**: the substitution-body test on each join, so no critical pair is closed vacuously.

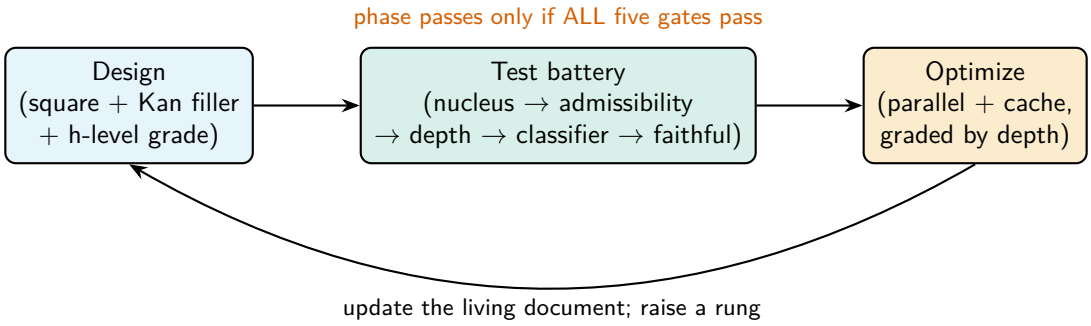


Figure 5: **The design loop we run each phase.** Design at the cell level, test with the five-gate battery (never a diamond alone), optimize what the certificate unlocks, then feed back: update this document and, when an instrument is ready, raise the ladder one rung.

Optimize: what certified confluence unlocks. Schedule-independence (from the nucleus) makes parallel and optimal-sharing reduction *sound*; the nucleus's free second run *is* normal-form caching; and the measured coherence depth tells the scheduler how aggressively it may reorder. The optimization is certified, and graded by the number the test produces — not a heuristic.

7 The destination: n-cell confluence

Why n-cell, and why now. Confluence has always been thought of as one property (do forks join). But once it is a *tower*, the higher rungs acquire *character*: at each level a different kind of mathematics governs whether the coherences cohere (Figure 6). The 2-cell is *logical* (do the two truth-values agree). The 3-cell is *algebraic* (do the two composites agree — the Gray/associativity coherence). The next is *topological* (does the filling exist — the Kan condition, a hole closed). The next *geometric* (do the fillers vary continuously — a connection, a transport). No rewriting platform has reached these, because none carried its coherence high enough to ask. Ours does, and Cubical HoTT supplies the instruments rung by rung. The program is to climb the ladder, one tested rung at a time, until confluence is a single graded invariant with four faces.

8 What we will do, and why

This is a *full maximal build*: every component below will be built to a working, runnable, non-vacuous state — there is no MVP, no demonstrator, no checkpoint to pause at. Verification runs *concurrently*, proving each landed piece real so the optimizations are sound; it never throttles the build. A missing instrument is *constructed*, never deferred; the only halt is a component proven impossible. The detailed project instructions live alongside this document; the components are:

1. **The certified core.** Represent *every* catalogued Boundary critical pair as a cubical `Square`; build each admissible join as its `hcomp` filler; prove each idempotent (the nucleus). *Why first:* this is the certificate layer the whole runtime stands on — every later optimization is only as sound as it. *Gate:* the five-gate battery over the full catalogue, the illegal-join control carrying no filler.

2. **The grade.** Wire the h-level instrument to read each net's coherence depth and pin every catalogued system's rung. *Why:* the depth *is* the scheduler's reorder budget. *Gate:* measured depth matches the known level for all, and a degraded system reads a lower rung.

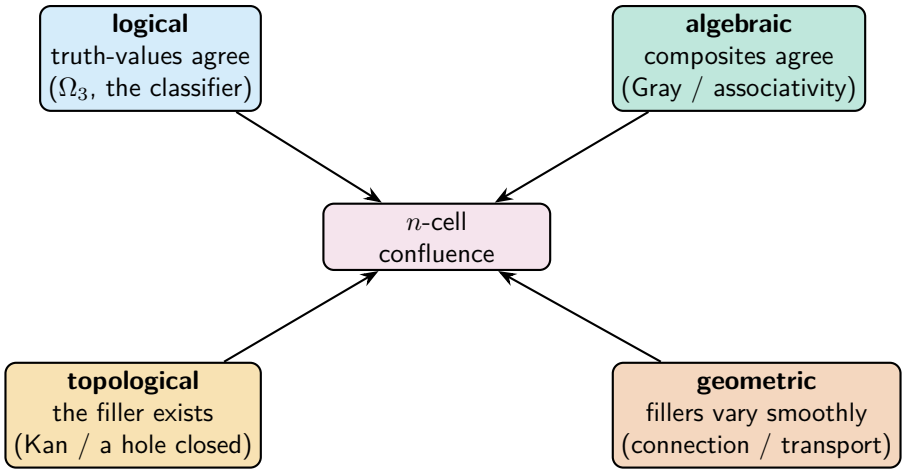


Figure 6: **Act 3 — the landscape.** As one invariant graded by coherence depth, confluence has up to four faces. The diamond sees only the logical face. Three of the four have a defined cubical instrument (logical: the Ω_3 classifier; algebraic: Gray coherence; topological: the Kan filler). The *geometric* face — "do the fillers vary coherently" — is drawn here to mark the territory, but it does *not* yet have a concrete target: defining what its witness is (and what non-instance it must exclude) is itself an open problem, not a checklist item. We claim a face only when a non-instance is correctly excluded; a name on a corner is not a face.

3. **The optimized runtime** (the applied payoff). Build the real reducer: parallel reduction (sound by the causal-independence certificate $\otimes_C$), normal-form caching (correct by the nucleus), scheduling graded by coherence depth, run on real Boundary nets. *Why:* this is the platform you run — faster, provably. *Gate:* a *measured* speedup over serial *with* a witness that the parallel result equals the serial normal form (fast *and* sound, not merely fast).

4. **The distributed substrate.** Model the typed open net as a fibration, reduction as transport, docking as a Kan lift. *Why:* reduction across machines, the open / horn story made runnable. *Gate:* torn-then-docked equals the whole-net normal form; an incompatible dock does not fill.

5. **The full n-cell tower.** Construct *all four faces* — logical, algebraic, topological, geometric — up to the rulial top, *building* any missing instrument rather than deferring it. *Why:* this is the new territory, the whole point. *Gate:* each face carries a positive witness *and* a correctly-excluded non-instance, none vacuous, the four composing into one graded invariant.

6. **Open the horizontal axis.** Add the multiway/branchial instruments (Section 4): a generic typed rule with its confluence-preserving constraint, the causal-edge lemma-selection heuristic, and embedded token-passing read-back. *Why:* this is the second axis — where genericity, side effects, and measurement live — without which the plane is only its vertical edge. *Gate:* a generic rule that preserves WCR[1] is admitted and one that opens a branch is correctly routed to the branchial (not silently accepted as confluent), and the branch-then-collapse diagonal reproduces the measurement of the quantum substrate.

Sources triaged into this revision. This map was sharpened against a reading of the multiway-rewriting and adjacent literature. From the Wolfram-model multiway program (Gorard et al., 2020, 2021) we take the horizontal axis: the branchial graph as superposition, **MuCauGraph** as the concrete weak-2-category model, the partial product $\otimes_C$ as the scheduler's reorder budget, and

causal lemma selection. From generic typed rules (Jiresch and Gramlich, 2011) we take higher-order matching with confluence-preserving constraints and a port-type discipline — the mechanism of the horizontal axis and a ready-made gate. From token-passing optimal reduction (Salikhmetov, 2016) we take strategy-independent normal form and embedded read-back. And two independent arrivals corroborate the geometry: quantum logic by gluing classifier squares without Hilbert space (Gunji et al., 2026), and a four-state cyclic "retention" algebra that reproduces the re-entry $\sqrt{-1}$ (Fellouri and Adjailia, 2026).

Each stage ends with this document updated — the wiring diagrams extended, the gate results recorded — so the map always matches the territory. The diamond told us the order of work does not change the answer. The plane tells us *how deeply* that is true when forks join, certifies *why*, measures *how far sideways* they spread when they do not — and shows us how much of the territory there is still to map.

References

Abdelkrim Fellouri and Meriem Adjailia. $\mathcal{K}_4$-core: A temporal semiring for hysteretic state computing. *preprint*, 2026.

Jonathan Gorard, Manojna Namuduri, and Xerxes D. Arsiwalla. ZX-calculus and extended hypergraph rewriting systems I: A multiway approach to categorical quantum information theory. *arXiv:2010.02752*, 2020.

Jonathan Gorard, Manojna Namuduri, and Xerxes D. Arsiwalla. Fast automated reasoning over string diagrams using multiway causal structure. In *Applied Category Theory (ACT)*, 2021. arXiv:2105.04057.

Yukio-Pegio Gunji, Kyoko Nakamura, Kazuto Sasai, Iori Tani, Mayo Kuroki, Alessandro Chiolerio, Andrew Adamatzky, and Andrei Khrennikov. Interplay between vertical and horizontal schemes of computation: From bayesian inference to quantum logic via gluing boolean algebras. *Entropy*, 28(5):498, 2026.

Eugen Jiresch and Bernhard Gramlich. Realizing monads in interaction nets via generic typed rules. Technical Report E1852-2011-01, TU Wien, 2011.

Anton Salikhmetov. Token-passing optimal reduction with embedded read-back. In *Computing with Terms and Graphs (TERMGRAPH)*, EPTCS 225, pages 45–54, 2016.