Retrieval as Belief Propagation

A Typed-Graph Architecture for Coherent, Auditable Reasoning in Discretionary Investment Agents

Theseus Technologies — Research Note

July 2026

Abstract. We describe an architecture for grounding an autonomous investment agent's decisions in an explicit, internally consistent set of beliefs. The core idea is to treat retrieval as a physical process: a query injects activation into a typed graph of principles, that activation spreads along weighted edges and decays with distance, and edges marked as contradictory carry *negative* conductance — so that incoherent branches of reasoning cancel themselves out before they ever reach the model. What survives is a small, ordered, self-explaining chain of beliefs, which is then grounded in market data and passed to a forecasting simulation. We situate this design against its nearest intellectual relative, Thagard's connectionist theory of explanatory coherence, against which it is a novel application rather than a first invention; we compare it to two alternative architectures — an unconstrained neighborhood-retrieval variant and a self-play reinforcement-learning variant — and we are explicit about which parts of the design rest on solid evidence, which rest on recent and largely unreplicated results, and which rest on no evidence at all. We close with the open research questions that determine whether this approach is viable.

Contents

1 Introduction

The premise of this work is a claim about agent design, not about markets: *an agent whose reasoning is drawn from an internally consistent set of beliefs will outperform one whose reasoning is not*, holding the quality of the underlying beliefs constant. A large language model given an unstructured pile of context cannot easily tell which parts of that context agree with each other; if the agent's knowledge base contains both a value-investing principle and a momentum principle that contradict it in a given situation, dumping both into a prompt does not surface the tension, it buries it. The architecture below is an attempt to make coherence a computed property of retrieval itself, rather than a separate check bolted on afterward.

The organizing metaphor is that **the context window is RAM; the belief graph is disk**. The graph holds everything the firm believes; each retrieval decides what small, coherent slice gets loaded into working memory for one decision. This is not a novel metaphor in retrieval-augmented generation generally, but it motivates a specific design choice: retrieval should be a *traversal* with its own dynamics (activation, decay, cancellation), not a similarity lookup followed by concatenation.

> *Claim:* a coherent chain of principles, retrieved by a process that lets contradictory beliefs cancel each other, grounded in real data, and projected forward by a simulation, produces better-justified investment decisions than either (a) an LLM reasoning over an undifferentiated context window, or (b) a model with no explicit belief structure at all.

This claim is plausible and is consistent with a body of retrieval-augmented-generation and cognitive-science literature discussed in §2, but it has not been tested end to end for this domain, and the weakest link — whether a short natural-language query can be reliably decomposed into the graph coordinates the system needs (§9) — is unproven. We flag evidentiary strength throughout rather than presenting the design as settled.

2 Related Foundations

The architecture combines ideas from three largely separate literatures that, as far as we are aware, have not previously been put together this way for financial reasoning. Being explicit about lineage matters here: it lets us borrow known failure modes instead of rediscovering them by trial and error.

2.1 Spreading activation in semantic memory

The mechanism in §4 — inject activation at seed nodes, let it spread along edges, decay with distance, gate on a firing threshold — is a direct descendant of Collins and Loftus's spreading-activation theory of semantic processing [1], developed to explain human semantic priming. That model was never intended as a retrieval algorithm for machines; its adaptation into knowledge-graph RAG systems is recent (§2.3).

2.2 Coherence as constraint satisfaction — the closest prior art

The single most important design feature of this architecture — that a **contradiction edge carries negative current, so surviving activation IS a coherence score** — is not new. It is, essentially, Paul Thagard's ECHO model of *explanatory coherence* [2, 3]: a connectionist constraint-satisfaction network in which propositions are units, mutually supporting propositions get *excita-*

tory links, contradictory propositions get *inhibitory* links, and the network is allowed to settle; the resulting stable activation pattern is Thagard's formal definition of which explanation is most coherent. McClelland's earlier interactive-activation-and-competition models [4] use the same excitatory/ inhibitory settling dynamic for perceptual constraint satisfaction (the "Jets and Sharks" model).

This is worth stating plainly because it changes how the coherence feature should be pitched: it is not a novel algorithm invented for this project, it is a thirty-five-year-old, well-studied computational theory of coherence, applied here to financial beliefs instead of scientific explanations. That is a genuine strength — it means the settling dynamics, failure modes (e.g. sensitivity to edge weights, multiple stable states), and evaluation methodology are already characterized in the literature and can be borrowed rather than reinvented. Prior work in this tradition should be cited as such, both for intellectual honesty and because reviewers or technical partners who know cognitive science will recognize it immediately.

2.3 Knowledge-graph retrieval versus vector RAG

The most direct precedent for using a typed knowledge graph as an LLM's retrieval substrate, with activation spreading over it, is SA-RAG [5], which reports large correctness gains over naive vector RAG on an automatically-constructed graph. Two additional data points are commonly cited in favor of graph-grounded retrieval for financial tasks: a Neo4j benchmark showing knowledge-graph retrieval sharply outperforming vector search on a financial-analysis task, and a GraphRAG evaluation on FinanceBench reporting fewer hallucinations and lower token cost than conventional RAG [6].

Evidentiary caution. These three results are frequently invoked together as if they constitute a literature; they do not, yet. SA-RAG is a preprint roughly seven months old at time of writing, revised once, with no independent replication. The Neo4j figure is vendor benchmark content published by a company that sells graph databases, which does not make it false but does mean it was not produced under adversarial or independent review. The FinanceBench result is a workshop paper at a co-located workshop (GenAIK), not a main-track result. None of this means the direction is wrong — graph-grounded retrieval for schema-heavy, KPI-dense domains like finance is a reasonably well-motivated hypothesis on priors alone — but the numbers ("+39% absolute", "∼56% vs ∼17%", "∼80% fewer tokens") should not be repeated externally as established fact. They should be treated as a promising but thin evidence base that this architecture would need to reproduce on its own data to actually rely on.

3 System Architecture: The Path-Tracking Pipeline

```
0  QUERY  (a situation / question arrives)
       |
       v
1  DIMENSION RESOLUTION          [stateless inference pass]
       - rank which domains of belief the query touches
       - assign an energy/resistance coefficient to each
       |
       v
2  SEED                          [1..N entry points]
       - inject activation at matched principles + the
         querying agent's own worldview anchors
       |
       v
3  SPREADING ACTIVATION          [over the typed principle graph]
       - current flows along typed edges, decays per hop
       - firing threshold acts as a noise gate
       - CONTRADICTS edges carry NEGATIVE current
       |
       v
4  PRINCIPLE PATHWAY  <-- the retrieved context window
       - the lit, connected chain of principles
       - pre-ordered and self-explaining: the trace IS the justification
       |
       v
5  DATA OVERLAY                  [empirical grounding]
       - quantitative extraction pulls relevant market data
       - layers real numbers onto the principle pathway
       |
       v
6  FORECASTING SIMULATION        [agent-based / Monte Carlo]
       - projects the grounded situation over an investment horizon
       |
       v
7  DECISION
       - price entry / exit, time entry / exit
```

Stages 1–4 are the *reasoning* half: what does the firm believe about this situation, and does that belief survive contact with itself? Stages 5–7 are the *empirical/predictive* half: what do the numbers say, and what happens next? This separation matters because the two halves have very different risk profiles — the reasoning half is a retrieval and consistency problem with reasonable precedent (§2); the predictive half is a forecasting problem in a genuinely hard, arguably unsolved research area (§6).

An important consequence of the design worth stating explicitly: because CONTRADICTS edges remove current rather than merely being flagged, coherence is not a post-hoc filter applied to a retrieved set — it is computed *during* retrieval, as the same operation that decides what gets retrieved. This is the design's central claim to novelty over standard RAG, and it is inherited directly from the ECHO tradition described in §2.2.

4 Substrate: Why a Typed Graph, Not a Vector Space

The retrieved pathway in Stage 4 could in principle live on either of two substrates: a **typed graph**, where every edge has a labeled type and a weight, or a **weighted vector space**, where traversal cost is a function of which embedding dimensions a query crosses. The vector-space version is simpler to build. We nonetheless argue for the typed graph, for two reasons that are structural rather than a matter of taste:

- **Auditability.** Every lit node in a graph was reached through a labeled edge, so the trace itself is the explanation — a byproduct of retrieval rather than a document written after the fact. A nearest-neighbor result in vector space cannot say *why* it was returned, only that it was close. For a fund that needs to produce a diligence memo justifying a trade, this is not a minor convenience; it is closer to a product requirement.
- **Coherence requires typed contradiction.** The negative-current mechanism in §3 and §5 needs an edge that specifically means "these two beliefs conflict." A vector space has distance, not semantics; there is no native contradiction edge to give negative weight to. Coherence could be reconstructed as a separate module downstream, but that abandons the property that makes this design distinctive — that retrieval and coherence assessment are the same computation.

The dimension-weighted-traversal idea from the vector-space alternative is not wasted, however: it maps cleanly onto edge-type weights in the graph (conduct more freely along causal edges, resist along bookkeeping edges), so the tuning concept survives even though the substrate does not need to change to get it.

5 Coherence as an Emergent Property of Signed Activation

Restated precisely, because it is the feature most likely to be quietly lost as the system grows in scope: contradiction edges carry negative conductance; mutually supporting beliefs add current; contradictory branches cancel. The surviving activation of a branch *is* its coherence score, with no separate coherence checker required. This is, again, Thagard's constraint-satisfaction formulation of coherence (§2.2), not a new idea, but it has apparently not been applied to belief-grounded financial reasoning before, which is where the actual contribution of this architecture lies.

One parameter remains genuinely open and should be settled empirically rather than by preference: whether contradiction should be **soft** (dampens activation continuously, producing a rankable worldview-coherence index per thesis) or **hard** (cancels outright, pass/fail). Thagard's own ECHO implementations use continuous (soft) activation update rules, which is mild evidence in favor of soft polarity here, but the financial-domain behavior of either choice is untested.

6 The Predictive Layer: Simulation, Not Inference

This section corrects a category error that is easy to make when "forecasting" methods are discussed loosely. Two methods are commonly proposed for the forward-looking layer: simulated annealing and iterative inference loops. Neither is a forecasting method.

- **Simulated annealing is an optimization algorithm.** It finds a global optimum of a cost function via a cooling schedule. It can *calibrate* a model's parameters against historical data; it cannot project a market forward on its own. Proposing it as "the predictive layer" is a category error.

- **Inference loops** (iterative refinement, as used in diffusion and other computer-vision methods) describe a computational *mechanism* — repeated refinement toward a fixed point — not a forecasting model. They are a plausible engine for running a simulation forward, not a substitute for one.
- **The literature that actually addresses forecasting market futures is agent-based modeling (ABM) and complexity economics**, alongside Monte Carlo and Bayesian forecasting methods. Farmer and Axtell's survey of ABM in economics and finance is explicit that forecasting remains the field's open problem, not a solved one [7].

Recommendation: the predictive layer should be an agent-based or Monte Carlo simulation of the situation, with annealing or inference loops used only for parameter calibration inside that simulation, never as the forecasting mechanism itself. A single, reusable simulation engine, re-parameterized per query, is preferable to a bespoke model per prompt.

This is the least-proven stage of the entire pipeline, and it should be described that way internally and externally. ABM forecasting is an active, genuinely unsolved research area; the Farmer/Axtell survey names the forecasting problem as open precisely because no consensus method exists. Committing to "agent-based simulation" resolves the category error above, but it does not resolve the underlying difficulty — it correctly relocates the hard problem to where it belongs.

7 Alternative Architectures: A Comparative Design Space

The path-tracking pipeline in §3 is one point in a broader design space, not the only viable design. Two alternatives are worth characterizing rigorously rather than dismissing.

7.1 Cluster retrieval

Instead of a lit, decayed traversal, this variant resolves a query to a single strong seed vector, fixes a cluster radius and a size cap, and retrieves every node in that neighborhood — collapsing Stages 2–3 of §3 into one step, with no per-edge decay, threshold, or activation tuning.

Property	**Path-tracking (§3)**	**Cluster retrieval**
Traversal machinery	Central, heavily tuned	Not needed — no traversal to tune
Burden on query resolution	Ranks dimensions	Must nail a single start point and radius; the entire result rides on it
Dependence on embedding quality	Helpful	Make-or-break — a mediocre vector space returns a bad neighborhood
Output shape	Ordered, self-explaining trace	Unordered cloud of principles
Coherence arithmetic (§5)	Native	Lost by default; would need to be reconstructed as a separate module

Cluster retrieval is not simply a worse version of path-tracking; it is a real alternative that wins

under two specific conditions worth naming precisely, because they are testable claims rather than matters of preference: (1) the embedding space turns out to be excellent but the traversal algorithms of §3–§5 cannot be made to work reliably; or (2) a sufficiently large context window lets a model use an unordered cloud of principles about as well as a curated trace, at acceptable token cost. Absent evidence for either, the standing objection to cluster retrieval is the same one that motivates staged retrieval generally: loading an undifferentiated block of context up front tends to degrade a model's reasoning relative to a curated, ordered trace, and this variant forfeits the auditability and coherence-as-arithmetic properties that are this design's main claims to distinctiveness.

7.2 Self-play reinforcement learning

The second alternative discards explicit belief structure entirely: give the system data and market access with minimal constraints, have it trade with simulated capital against real or simulated markets, repeat at scale, and keep what wins.

This is the alternative most likely to be oversold, and deserves a direct critique rather than a pros/cons list. Self-play works spectacularly for games like Go and chess because the opponent's rules are fixed and the environment is stationary and fully specified by the game's own state. Financial markets are neither: the "opponent" (aggregate market behavior) is itself adaptive, regimes shift on timescales shorter than a training run, and a policy that discovers an exploitable pattern in historical or simulated data has no guarantee that pattern persists once capital is deployed against it — an instance of Goodhart's law as much as a machine-learning problem. This is a known, structural limitation of applying self-play RL to markets, not merely an engineering risk to be managed with more compute. Absent a specific argument for why this particular system's environment is stationary enough for self-play to transfer, the approach should be treated as a high-variance research bet, not a credible path to a production trading signal.

The approach also produces an interpretability gap: an emergent policy may not yield the auditable trace that is this project's main product differentiator (§4), and compute cost is unbounded without an explicit cap (§8). It is best framed as a fenced, budget-capped research program run in parallel to the core architecture, evaluated on whether it discovers anything that a hand-built ontology would not have — not as a competing production path.

8 Computational Cost as an Architectural Property

Cost is not incidental to this comparison; it is a structural property that differs by architecture in ways worth stating precisely, at order-of-magnitude precision only.

Driver	**Path-tracking**	**Cluster**	**Self-play**
Graph storage / traversal	Low (CPU, cheap)	Low	N/A
LLM tokens per query	Low–medium (small curated trace)	Medium–high (large context cloud)	N/A at inference time
Simulation compute	Medium (CPU-bound, bursty)	Medium	—

Driver	Path-tracking	Cluster	Self-play
Training / self-play GPU	—	—	Very high, continuous
Cost predictability	Predictable	Scales with context size	Unbounded without an explicit cap

The qualitative read-through: path-tracking is the cheapest and most predictable architecture at scale, because small curated traces produce small token bills. Cluster retrieval looks cheaper to build but its token cost grows with context size and can quietly exceed path-tracking's cost once usage scales. Self-play sits in a different cost regime entirely and should be evaluated as a bet with an explicit stop-loss on spend, not as a line item comparable to the other two — consistent with the critique in §7.2 that its underlying premise (a sufficiently stationary market to learn against) is itself unproven.

9 Open Research Questions

The following are unresolved technical questions that determine whether this architecture is viable, stated as research questions rather than as items for a status meeting.

1. **Query decomposition validity.** Can a short natural-language query be reliably decomposed into a weighted map over belief dimensions (§3, Stage 1) at all? This is the load-bearing assumption of the entire pipeline and is currently unvalidated; no cited evidence (§2) addresses this specific subproblem, since SA-RAG and the graph-RAG benchmarks all assume the query is already usable as a retrieval key, not that it must first be decomposed into domain weights.
2. **Polarity.** Soft versus hard contradiction (§5) is untested in this domain; Thagard's own implementations favor soft/continuous updates, which is weak prior evidence, not domain evidence.
3. **Seeding.** Single-seed versus multi-seed activation, and default decay/firing-threshold values, remain to be determined empirically rather than assumed.
4. **Predictive method.** Confirming agent-based/Monte Carlo simulation (§6) as the forecasting engine resolves a category error but not the open forecasting-methodology problem itself.
5. **Extraction.** Two distinct extraction problems are bundled under "building the graph": (a) typed principle nodes and edges with confidence and provenance, and (b) quantitative data extraction for Stage 5. SA-RAG's use of an automatically-constructed graph is some evidence that (a) does not require a perfect hand-curated ontology, but this has not been verified on financial belief content specifically.
6. **Coherence-empirical weighting.** How much weight should the pipeline give to philosophical/principle-based reasoning versus purely empirical pattern-matching (the self-play extreme in §7.2)? This is a genuine, unresolved research question about the architecture's design point, not merely a preference.

10 Glossary

- **Principle pathway** — the lit, connected chain of belief nodes returned by Stages 1–4 of §3; the retrieved context window.

- **Coherence index** — the total surviving activation of a branch under soft polarity (§5); Thagard's coherence value, computed for this domain.
- **Substrate** — the underlying store the retrieval process runs on: typed graph versus vector space (§4).
- **Energy/resistance coefficient** — the edge-type weights and per-query conductance values that tune how activation flows through the graph.
- **Simulation engine** — the single, reusable agent-based/Monte Carlo forecasting model of §6, re-parameterized per query; deliberately distinguished from "model" in the neural-network sense.

11 Conclusion

The contribution of this design is not the invention of spreading activation or of coherence-as-constraint-satisfaction — both have decades of precedent in cognitive science (§2) — but the specific application of that machinery to grounding an investment agent's reasoning in an auditable, internally consistent belief graph, with a clean separation between the reasoning half (Stages 1–4) and the empirical/predictive half (Stages 5–7). The reasoning half rests on a reasonably credible, if still thin and recent, evidence base (§2.3) and a well-established cognitive-science precedent (§2.2). The predictive half rests on a genuinely open research problem (§6) that no amount of engineering discipline resolves on its own. The two alternative architectures (§7) are not weaker versions of the same idea; they occupy different, definable points in a real design space, one of which (self-play) carries a structural objection — market nonstationarity — serious enough that it should be treated as a bounded research bet rather than a production candidate. The single highest-leverage next question is not architectural but empirical: whether a query can actually be decomposed into the structure Stage 1 assumes it can (§9, item 1). Everything downstream depends on that being true.

References

[1] A. M. Collins and E. F. Loftus, "A spreading-activation theory of semantic processing," *Psychological Review*, 82(6), 407–428, 1975.

[2] P. Thagard, "Explanatory coherence," *Behavioral and Brain Sciences*, 12(3), 435–502, 1989.

[3] P. Thagard and K. Verbeurgt, "Coherence as constraint satisfaction," *Cognitive Science*, 22(1), 1–24, 1998.

[4] J. L. McClelland and D. E. Rumelhart, "An interactive activation model of context effects in letter perception: Part 1," *Psychological Review*, 88(5), 375–407, 1981.

[5] Pavlović et al., "Leveraging Spreading Activation for Improved Document Retrieval in Knowledge-Graph-Based RAG Systems" (SA-RAG), arXiv:2512.15922, Dec. 2025 (v3 May 2026).

[6] "GraphRAG on FinanceBench," ACL 2025 GenAIK Workshop (COLING); see also Neo4j, "Knowledge graph vs. vector RAG: Benchmarking, optimization levers, and a financial analysis example"; and Microsoft Research's GraphRAG framework (Larson, Truitt et al.).

[7] R. Axtell and J. D. Farmer, "Agent-Based Modeling in Economics and Finance: Past, Present, and Future," *Journal of Economic Literature*.

[8] "LLM-empowered knowledge graph construction: A survey," arXiv:2510.20345, Oct. 2025.