Project Overview: Hierarchical Schema-Guided Agentic RAR System

Vision

This project aims to build a modular, fully self-hostable, and schema-evolving knowledge ingestion and retrieval pipeline that combines the strengths of: - **Agentic Reasoning-Augmented Retrieval (RAR)** for symbolic ingestion - **Hierarchical Document Structuring** for multi-level context representation - **Schema Evolution and Deduplication** for long-term scalability - **Community Detection and Clustering** for symbolic refinement - **Graph-Aware Embedding Strategy** for reasoning-informed vector search - **Curriculum Generation via Knowledge Graphs** for fine-tuning future domain-specific reasoning agents

The system supports **domain-agnostic ingestion**, **self-improving schema evolution**, and **query-time reasoning** with dynamic agent orchestration. It will operate on structured (symbolic), semi-structured (markdown, HTML), and unstructured (PDF, OCR) content.

This architecture supports research publication by comparing the system against GraphRAG, PathRAG, LightRAG, HiRAG, AutoSchemaKG, GEARs, QMKGF, and RAR using quantitative retrieval performance and qualitative structure metrics. The system is also conceptually aligned with and extends the insights from the 2025 Harvard/Princeton "Bottom-Up Domain-Specific Superintelligence" paper.

Emerging lines of research that may enhance future components include: - **GNN-enhanced topological embeddings** for capturing multihop symbolic structure - **Skim-RAG and SciSchema** for rapid schema extraction and universal schema distillation - **GRITformer, GAT, R-GCN** for graph-aware transformer integration - **Cross-domain Schema Merging** with convergence tracking and symbolic schema alignment agents

---

Architecture Summary

1. Schema Seeding and Evolution Loop

- Start with a small set of universal schema roles (e.g., `Claim`, `Method`, `Evidence`, `Section`)
- Use LLMs to extract document structure and propose new schema roles
- Automatically deduplicate overlapping schema roles using clustering and semantic matching
- Continuously refine schema without human-in-the-loop using:
- Frequency thresholds
- Similarity scores
- Community structure feedback
- Cross-corpus convergence metrics

2. Hierarchical Document Chunking

- Documents are parsed into tiered chunks:

- `Document` → `Section` → `Paragraph` → `Sentence`
- This structure forms a scaffolding for symbolic graph building and agent task scoping
- Combines both **explicit hierarchy** (structure of the document) and **implicit hierarchy** (emergent through community formation and semantic linkage)

3. Expert Agent Swarm for Symbolic Ingestion

- Specialized LLM-driven agents (e.g., `ClaimExtractorAgent`, `ReferenceLinkerAgent`) ingest content and instantiate nodes/edges in Neo4j using the current schema
- Agents follow schema-defined roles to ensure consistency and reuse
- This feature distinguishes RAR-based systems from other symbolic pipelines

4. Community Detection and Semantic Clustering

- After symbolic graph construction:
- Detect communities using centrality, shared context, and role similarity
- Use these clusters to guide further schema deduplication and refinement
- Support schema convergence tracking across unrelated corpora (cross-domain)

5. Graph-Aware Embedding Layer

- Once symbolic construction is complete, nodes are embedded with optional context from schema roles, document lineage, or neighbor relations
- The system supports:
- **Initial full embedding post-KG**
- **On-demand Just-In-Time (JIT) embedding**
- **Topological-aware embedding roadmap** with GNN layers (GAT, R-GCN) and graph transformers (GRITformer)
- Embeddings are cached and versioned using chunk + schema role + embedding model ID

6. Query-Time Retrieval + PathRAG Traversal

- Query is parsed by LLM agents to determine target schema roles (e.g., `Why → CausalExplanation`)
- Use vector search + symbolic traversal to find relevant context
- Apply path optimization to find shortest semantic chains across communities
- Response generated with graph trace for explainability

Distinction From Prior Systems (Summary Table)

System	Schema Awareness	Hierarchy Support	Agent Specialization	Deduplication Strategy	Retrieval Type	Vector Use Strategy
GraphRAG	Implicit	None	No	No	Graph-based	Pre-embedded

System	Schema Awareness	Hierarchy Support	Agent Specialization	Deduplication Strategy	Retrieval Type	Vector Use Strategy
PathRAG	No	None	No	No	Path traversal	Pre-embedded
LightRAG	No	2-level (shallow)	No	Yes	Multi-tier	Pre-embedded
HiRAG	Partial	Explicit	Some	Partial	2-Level RAG	Mixed
AutoSchemaKG	Fully Autonomous	Implicit only	None	Yes	KG Expansion	Embeds post-KG
RAR (baseline)	Predefined or Tool-Guided	No	Yes (core feature)	Not specified	Symbolic Reasoning	Query-late or manual
GEARs	Manual and Task-Aligned	Implicit	Yes (task routers)	Not the focus	Multi-hop KG QA	Transformer + KG-enhanced
QMKGF	Question-Aligned	Question Type-Based	Yes (QType classifiers)	Question-Conditioned	KG + QA fusion	Query-specific KG selection
Bottom-Up Superintelligence	Fixed and Curated	Domain-specific (manual)	No	Manual/ curated	KG → Curriculum Tasks	N/A (used for fine-tuning)
Our System	Evolving and Adaptive	Explicit + Implicit	Yes (modular agents)	Iterative & Convergent	Hybrid Retrieval	Full + Optional JIT + GNN-Ready

Additional conceptual distinctions: - **Schema Convergence + Adapter Detection**: Enables symbolic translation across corpora, beyond AutoSchemaKG's local schema proposals - **Agentic Curriculum Generation**: Inspired by Bottom-Up Superintelligence, but includes graph traversal agents that map knowledge paths to fine-tuning tasks - **Topological Embedding Versioning**: Allows symbolic + structure-aware hybrid representations for future retrieval tasks

---

Research Outcomes

This architecture is designed to support experimental comparisons with: - **Accuracy of retrieval** across question types (why, what, how, who) - **Graph coherence** (node redundancy, path density, interpretability) - **Schema growth/efficiency** (rate of new schema convergence, reduction in manual schema additions) - **LLM agent performance** (time-to-ingest, role adherence, collision detection) - **Query speed and token efficiency** across semantic vs symbolic vs hybrid traversal - **Performance of graph-aware embeddings vs**

node-only embeddings in downstream retrieval - **Effectiveness of curriculum task generation from symbolic graphs** for fine-tuning or evaluation - **Impact of schema role-conditioned embeddings** vs vanilla chunk-based embeddings - **Effectiveness of schema distillation agents** in reducing schema entropy over large corpora

Baseline systems for comparison: - HiRAG (hierarchy-focused) - AutoSchemaKG (schema induction-focused) - GraphRAG (graph retrieval baseline) - LightRAG (deduplication baseline) - PathRAG (optimized retrieval baseline) - RAR (symbolic-first, agent-driven reasoning baseline) - GEARs (task-based KG traversal baseline) - QMKGF (question-structured KG fusion baseline) - Bottom-Up Domain Superintelligence (KG-driven curriculum for fine-tuning baseline) - GRITformer / GAT / R-GCN (GNN/topological reasoning baselines) - Skim-RAG / SciSchema (schema induction and compression baselines)

Next Steps for Architecture Agent

1. **Operationalize Schema Layer**: Define initial schema roles and versioning strategy
2. **Design Chunking Engine**: Support multiple formats and hierarchical layers
3. **Agent Runtime Plan**: Modularize ingestion agent types and schema access patterns
4. **Graph Storage Strategy**: Determine edge types, traversal rules, and community logic
5. **Evaluation Metrics & Logging Hooks**: Add instrumentation for retrieval and graph performance
6. **Deploy First Research Corpus**: Run on diverse document types to evaluate schema evolution
7. **Prepare graph-embedding upgrade pathway**: Assess when and how to apply topologically aware embeddings (e.g., GNN, GRITformer)
8. **Prototype curriculum generator**: Explore symbolic graph-to-task conversion for future model fine-tuning and benchmarking
9. **Incorporate schema adapter discovery agents**: Analyze convergence patterns across unrelated corpora
10. **Model schema entropy and convergence rate**: Use this as a heuristic for symbolic compression efficiency

All architecture decisions should be made with these research outcomes and comparisons in mind.