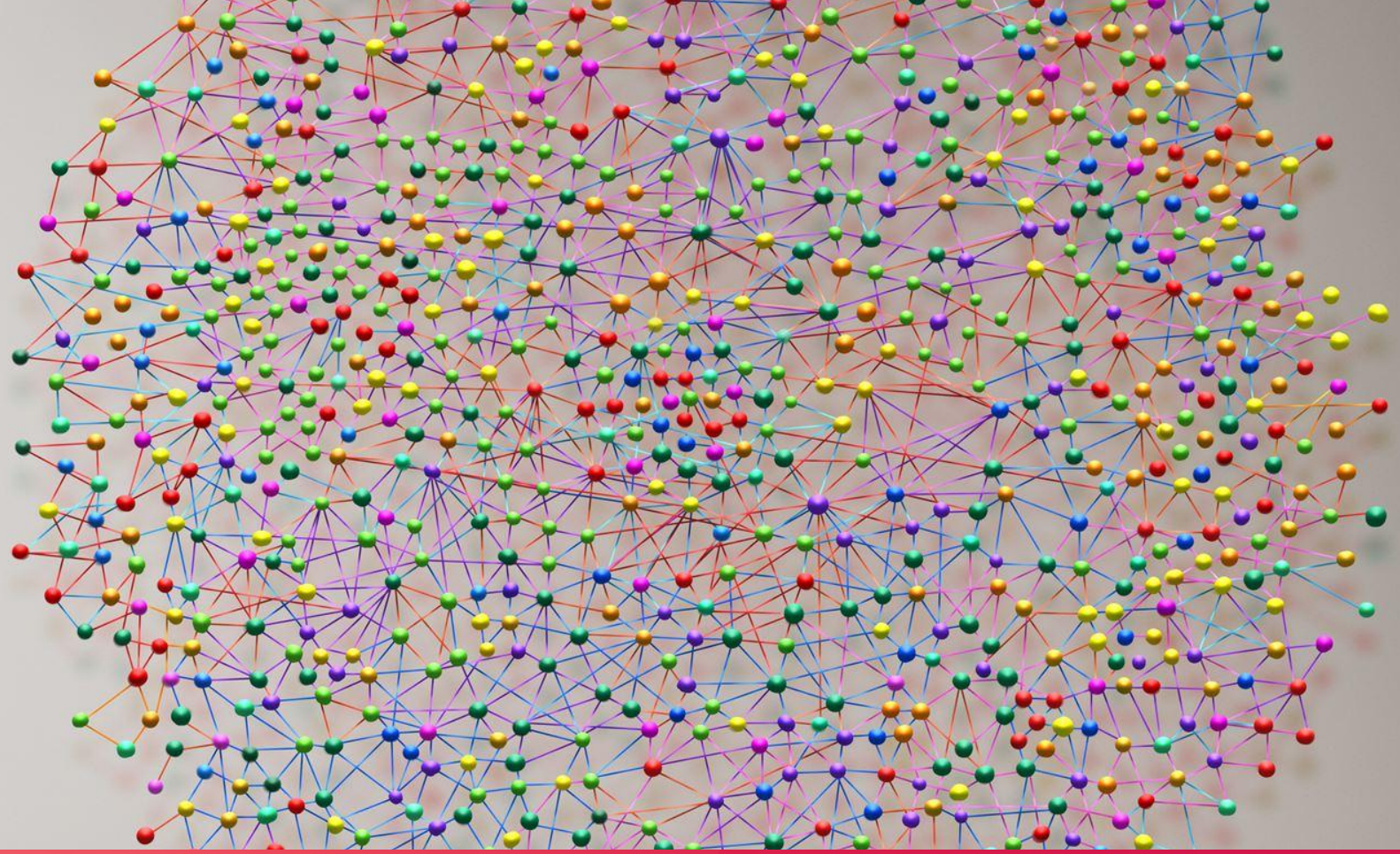


Designing and Implementing Knowledge Graphs in the Legal Domain

STARAM

STARAM

Staram's Objectives

Build legal knowledge graphs which ensure -

- Accurate Legal Answers

Deliver precise answers supported by verifiable citations from statutes and court judgments. Ensure AI-generated insights are explainable, transparent, and aligned with legal principles.

- Build Comprehensive Knowledge Graphs

Structure legislative and judicial data using legal knowledge graphs for enhanced AI-powered recommendations

- Scalable Compliant Workflows

Implement repeatable workflows that scale across jurisdictions while ensuring transparency and traceability.

What we have built till now…

- Structured Legislative Data Sources

Knowledge graphs (KGs) have been constructed on over 20,000 legislations with over 100 unique nodes and edges.

- FIRAC Encoding Methodology for Judgment KGs

FIRAC method structures legal information into facts, issues, rules, analysis, and conclusions for clarity. This has helped us maintain clean and relevant context on Neo4j servers.

- Knowledge Graphs for Explainable AI

Knowledge Graphs to represent entities such as citations, statutes, compliances and precedents in over 300,000 Supreme Court and High Court judgments.

Benefits of Legal Knowledge Graphs (LKGs)

1. Domain-Specific AI Expertise

Generic LLMs lack legal knowledge. The LKGs map statutes, case law, courts, and procedures, enabling AI to answer precise queries like "Find SC cases where cheque bounce convictions were overturned for lack of legally enforceable debt".

2. Unified Data Across Fragmented Systems

Legal teams juggle between various systems, DMS, databases, compliance trackers, billing tools, etc.—each with different terminology. LKG's reconcile these by defining canonical concepts (Matter, Case no., Contract, Obligation) and mapping local labels, enabling integrated queries across all systems.

3. Explainable & Audit-Ready Decisions

When AI flags a risky clause or recommends a position, ontologies trace the reasoning through explicit legal relationships—e.g., "Clause violates Section X of DPDP Act → RBI governance principle → risk."

4. Superior Legal Research via Knowledge Graphs

LKG's organize statutes, cases, and concepts with semantic relationships ("interprets," "overrules," "follows"), ensuring AI retrieves authoritative precedents and current legal positions rather than hallucinated or outdated answers.

Imagine a user asks: *"Show me liability clauses in active contracts with SAAS vendors that expire in 2025."*

How the system handles it:

Vector Database (Semantic Search): It scans thousands of pages to find paragraphs that *mean* "Liability" (even if they use words like "Indemnity," "Responsibility," or "Damages"). It handles the messy unstructured text.

Knowledge Graph (Structured Filter): It filters those results using explicit metadata:

Vendor_Location == "Germany" : Contract_Status == "Active" : Expiry_Year == "2025"

Final Result: The LLM receives only the relevant clauses from the *correct* contracts, ensuring it doesn't hallucinate a clause from an expired contract or the wrong country.

Why GraphRAG Matters

It solves the biggest problem of normal RAG:

Text chunks don't capture structured relationships. Graphs do.

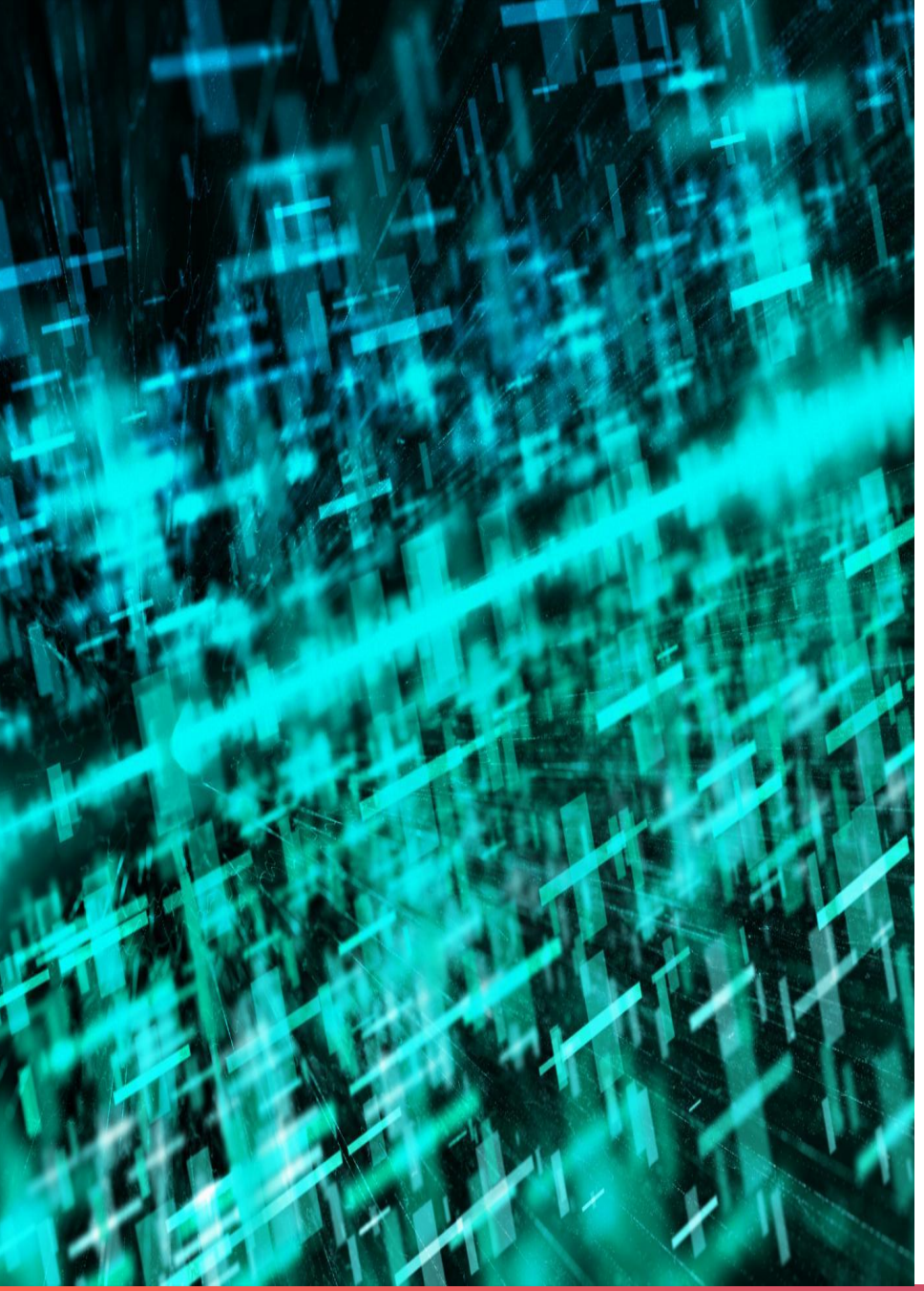
Problem	Traditional RAG	GraphRAG
Multi-hop questions	Weak	Strong
Explainability	Low	High
Hallucination	Higher	Lower
Entity grounding	Poor	Strong
Context stitching	Manual	Automatic

The Trade-off: Complexity

The single main trade-off we have to accept with this powerful combination is architectural complexity.

A GraphRAG system requires:

- An additional ingestion step to extract entities and relationships from unstructured data and map them into the Neo4j graph.
- Management and maintenance of a dual-retrieval pipeline (vector search + graph query).



Managing Context and Avoiding Failures

- Common Failures of LKGs

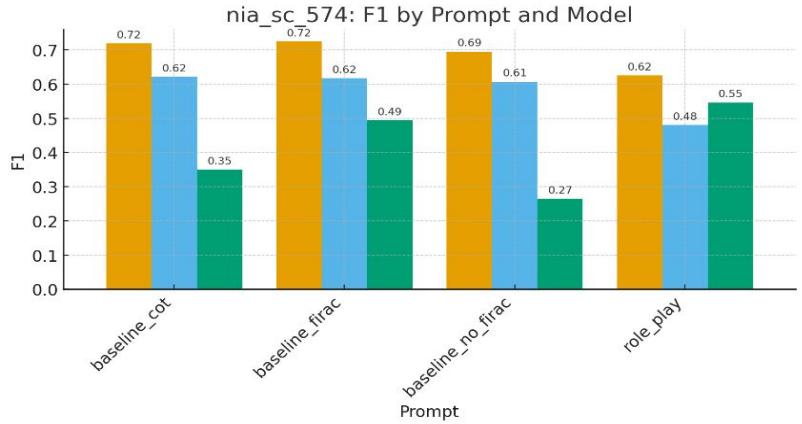
We addressed the four key issues of legal data –

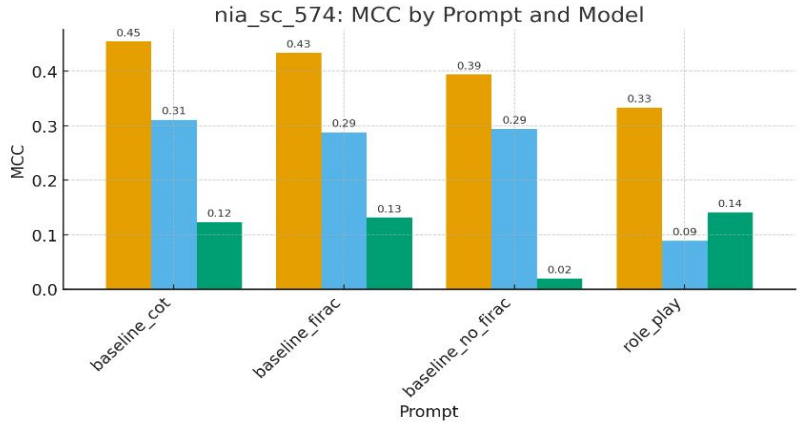
(i) context poisoning,
(ii) context distraction,
(iii) context confusion, and
(iv) context clash

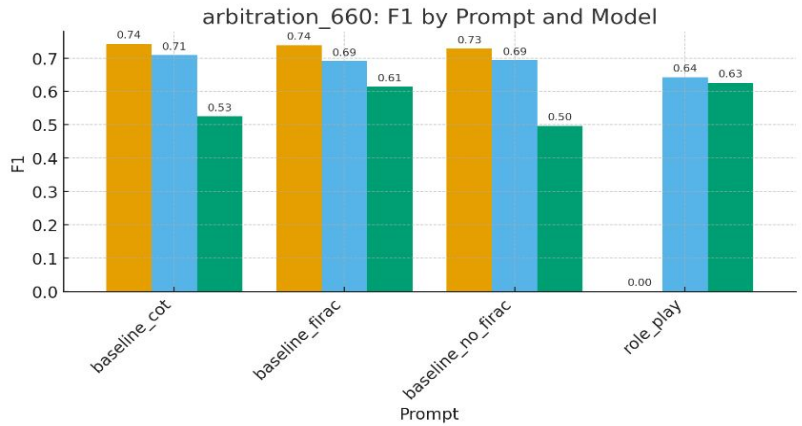
each affecting agent performance differently.

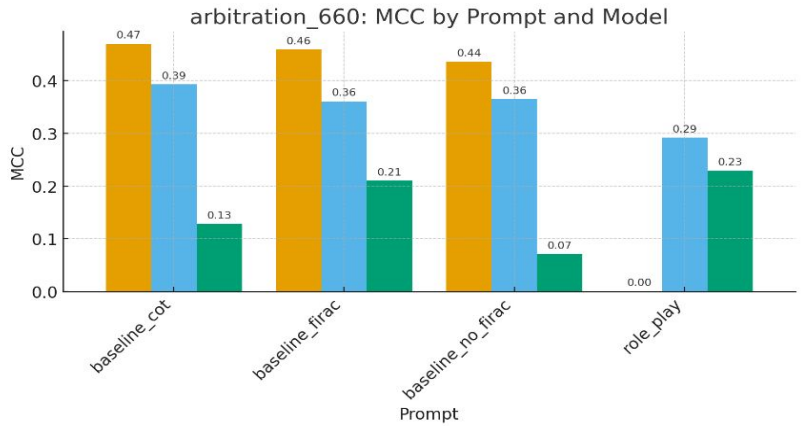
- Mitigation Techniques

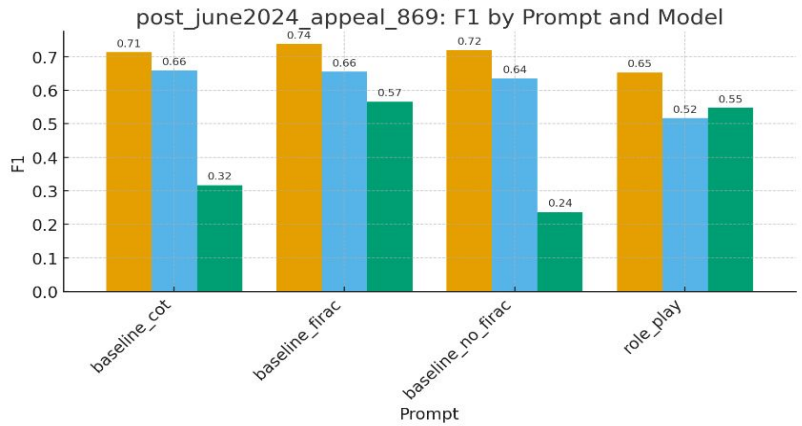
We use techniques like – content pruning through FIRAC extraction, summarization, and context quarantine to maintain clean and relevant context.

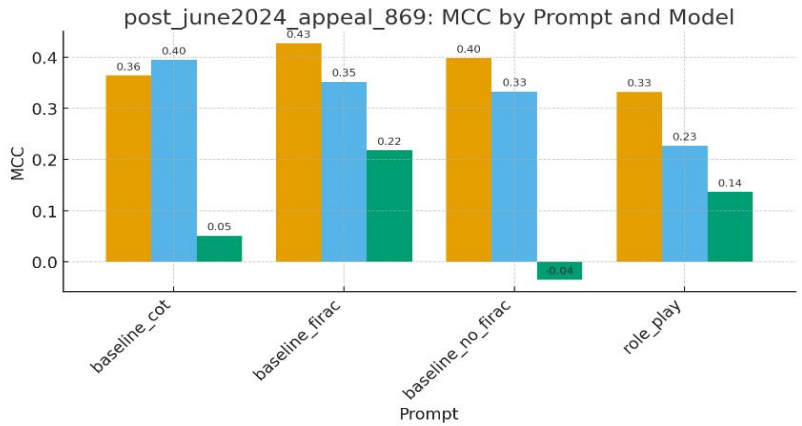


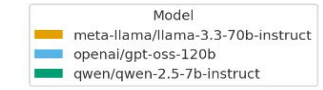












Glasgow Study

Our study with the team from the University of Glasgow showed that working with FIRAC actually **does work better** than plain CoT.

The results of the FIRAC study confirmed that lengthy case judgments led to failure modes like context poisoning, distraction, confusion and context clash. This impacted AI accuracy and agent performance.

We have reduced server and compute costs for storing over 2 TB of data significantly by adopting these methods…

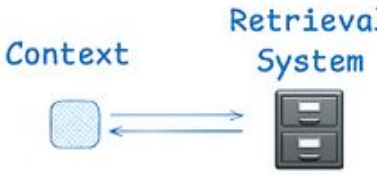
STARAM

Context Offloading

Context Offloading is the act of storing information outside the LLM's context, usually via a tool that stores and manages the data.

RAG

Retrieval-Augmented Generation (RAG) is the act of selectively adding relevant information to help the LLM generate a better response.

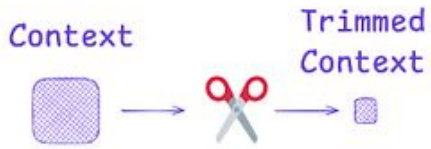


Tool Loadout

Tool Loadout is the act of selecting only relevant tool definitions to add to your context.

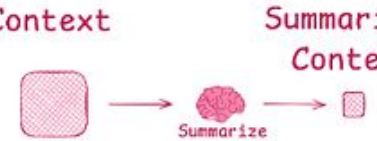
Context Pruning

Context Pruning is the act of removing irrelevant or otherwise unneeded information from the context.



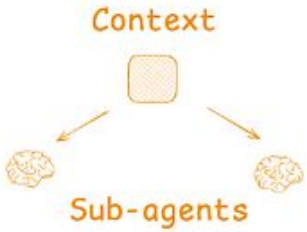
Context Summarization

Context Summarization is the act of boiling down an accrued context into a condensed summary.



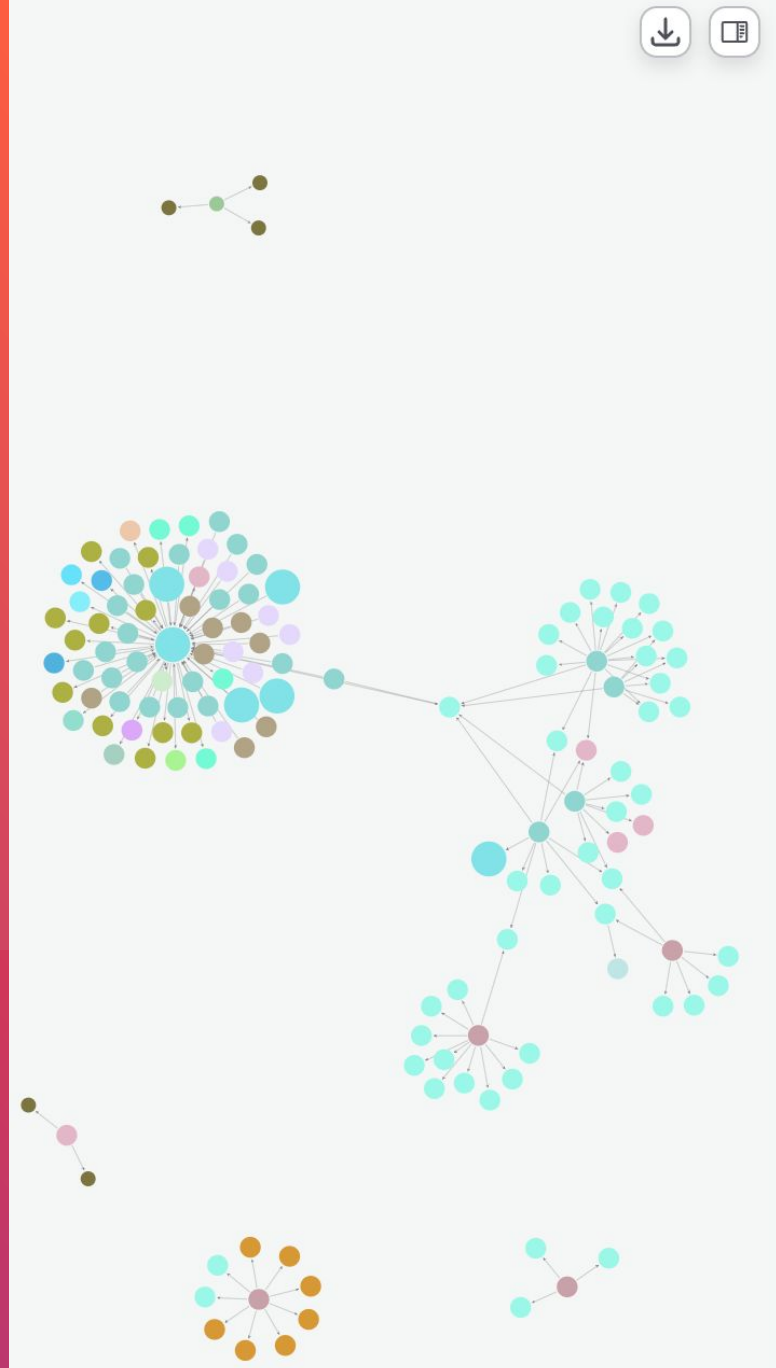
Context Quarantine

Context Quarantine is the act of isolating contexts in their own dedicated threads, each used separately by one or more LLMs.



KG DEVELOPMENT

- SAMPRITHA MANJUNATH



Results overview

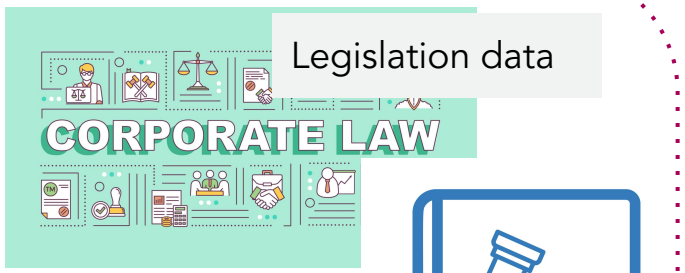
Nodes (165)

* (165) | Act (7) | Amendments_and_Repeals (1) | Applicability_of_Statute (1) | Case (25) | Circular (7) | Conditions_and_Prerequisites (1) | Definitions_and_Terms (1) | Eligibility_Criteria (1) | Enforcement_Mechanisms (1) | Judicial_Precedents (1) | Key_Dates (1) | Law (6) | Node (3) | Notification (7) | Paragraph (9) | Part (4) | Procedural_Steps (1) | Regulatory_Bodies (11) | Responsibilities_Specified_on_Key_Persons (1) | Rule (8) | Section (10) | Subchild (1) | Sublaw (52) | SubSection (5)

Relationships (166)

* (166) | CONTAINS (21) | HAS_NODE (21) | HAS_SECTION (3) | HAS_SUBCHILD (1) | HAS_SUBLAW (30) | PARENT (15) | REFERS_TO_ACT (22) | REFERS_TO_LAW (36) | REFERS_TO_SECTION (5) | RELATION (4) | UNDER (8)

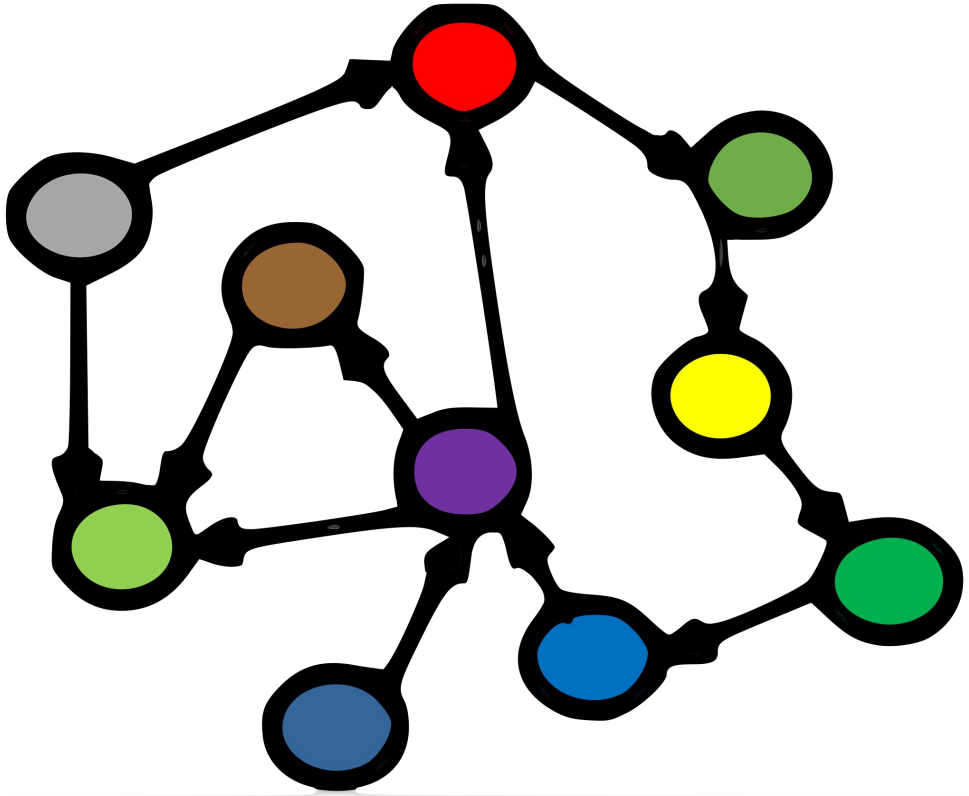
Source data and KG structure



Legislation data

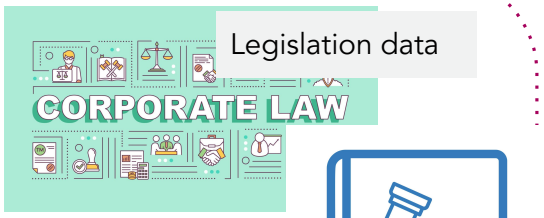
Case data

Legal documents



neo4j

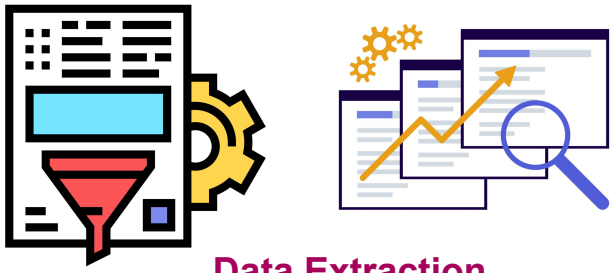
Source data and KG structure



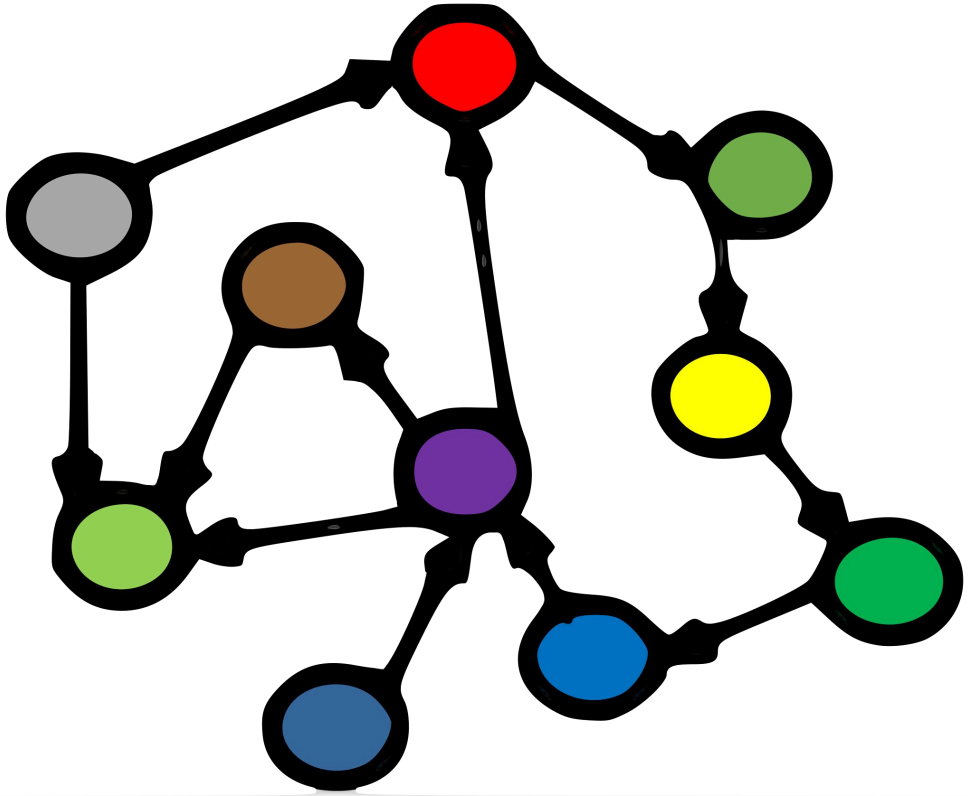
Legislation data

Case data

Legal documents



Data Extraction



neo4j

Knowledge Graph Demo

https://browser.neo4j.io/

STARAM

What is added?

Acts → Law

Acts → Part/Chapter → Sections → Subsections → Paragraph → Subparagraph

Rules → Sections → Subsections → Paragraph → Subparagraph
Rules → Acts

Notifications, Circulars, Trade Notices Act

Cases → Acts & Sections

Cases → Law

Compliance → Act

Additional Nodes → Act

What is extracted?

Additional nodes that add further context to the KG like deadlines, amendments, triggers, penalty etc., (20 additional types of nodes) are extracted from Act level.

Compliance data are extracted from Rules & Regulations data.

We have 8 types of compliance data extracted

- ❑ Date based
- ❑ Form based
- ❑ Event bases
- ❑ Checklist bases
- ❑ Regulator based
- ❑ Act based
- ❑ Context based
- ❑ Dependency based

Case treatments - positive & negative - are extracted from Judgement data

Multi-Agent GraphRAG System Architecture

Legal Query Types - Intelligent Routing

Type 1: Interpretation of Law

Understanding meaning, scope, or applicability of statutory provisions

Example Keywords:

- "What does Section X mean?"
- "Scope of Rule Y"

Output Focus:

- Statutory text
- Legislative intent
- Subsections analysis

Type 2: Judicial Precedents

Seeking case law and judicial interpretations

Example Keywords:

- "Cases on arbitration"
- "Supreme Court rulings on X"

Output Focus:

- Key judgments
- Legal holdings
- Case hierarchy

Type 3: Facts + Law Application

Applying legal provisions to specific factual scenarios

Example Keywords:

- "Given facts A, B, C, what is legal position?"
- "Can X do Y under this law?"

Output Focus:

- Fact-law synthesis
- Application analysis
- Relevant precedents

Type 4: Statutory + Case Law Synthesis

Complex queries requiring both statutory provisions AND their interpretation

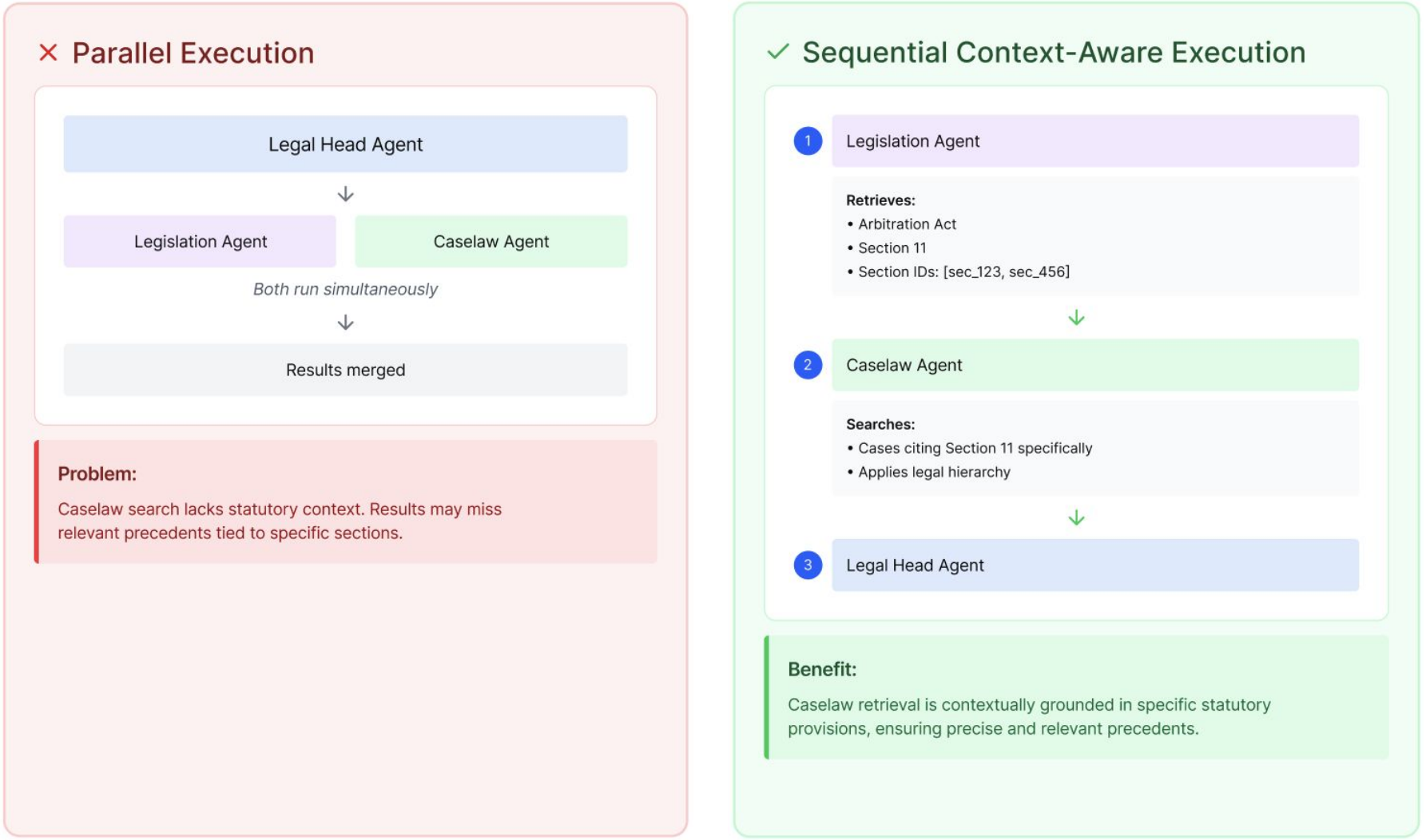
Example Keywords:

- "Section X and related judgments"
- "How have courts interpreted Rule Y?"

Output Focus:

- Statutory framework
- Case law interpretation
- Synthesized analysis

Where a query requires sequential context, we avoid parallel execution…



Scaling Graph Traversal with Matrix-Based Cache

The Challenge: Exponential Growth

2024	2025	2026
20K Acts	35K Acts	50K+ Acts
150K Cases	3M Cases	15M Cases

Problem:
Traditional Cypher query:
~2.3 seconds
at 300K+ nodes

Graph Relationship

Act
→
Section
→
Case

(3-hop traversal)

Mathematical Equivalent

Sparse Adjacency Matrices

HAS_SECTION × REFERRED_BY
=
Direct paths

Performance

Query time:
~40ms

57x
faster than Cypher

Why Sparse Matrices Work for Legal Graphs

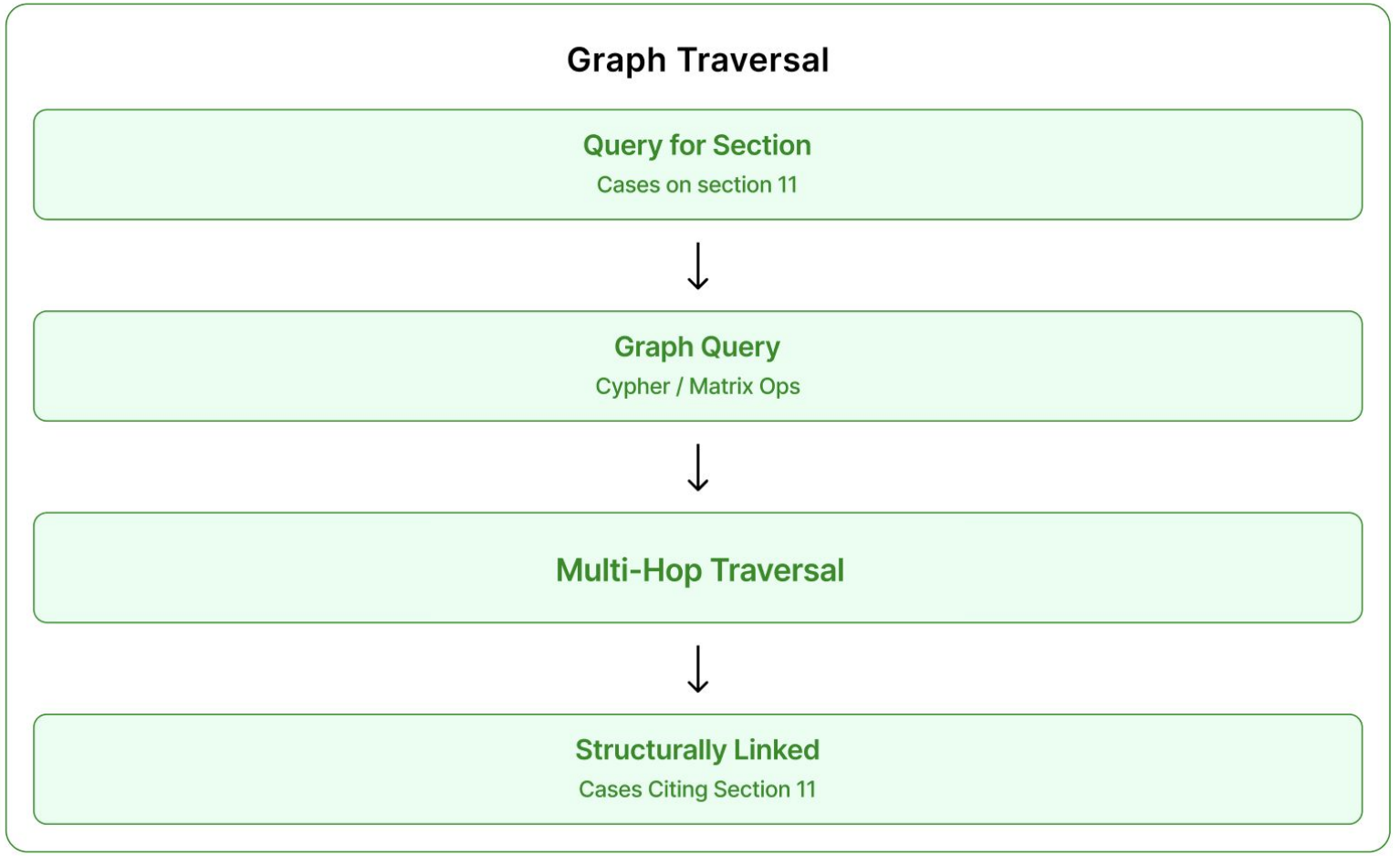
Legal knowledge graphs are inherently sparse:

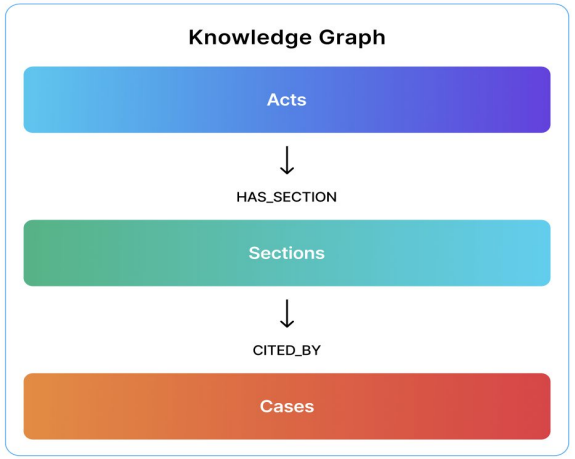
- Acts have 50-200 sections (not millions)
- Cases cite 5-15 sections (not thousands)
- Sections have 3-10 subsections (not hundreds)

Memory efficiency:

- Dense Matrix: 200K × 200K = 40B entries → 160 GB
- Sparse Matrix: ~400K connections → 50 MB
- **99.9% reduction**

Graph RAG for Structural Navigation





FIRAC Components

- **F** **Facts** — Case Circumstances
- **I** **Issues** — Legal Questions
- **R** **Rule of Law** — Legal Principles
- **A** **Analysis** — Application of Law to Facts
- **C** **Conclusion** — Final Judgement

Basis of Evaluation

A. Core content qualities

B. Reasoning and structure

C. Use of authorities and citations

D. Linguistic quality and usability

E. Safety, ethics, and compliance

Test Query 1 -

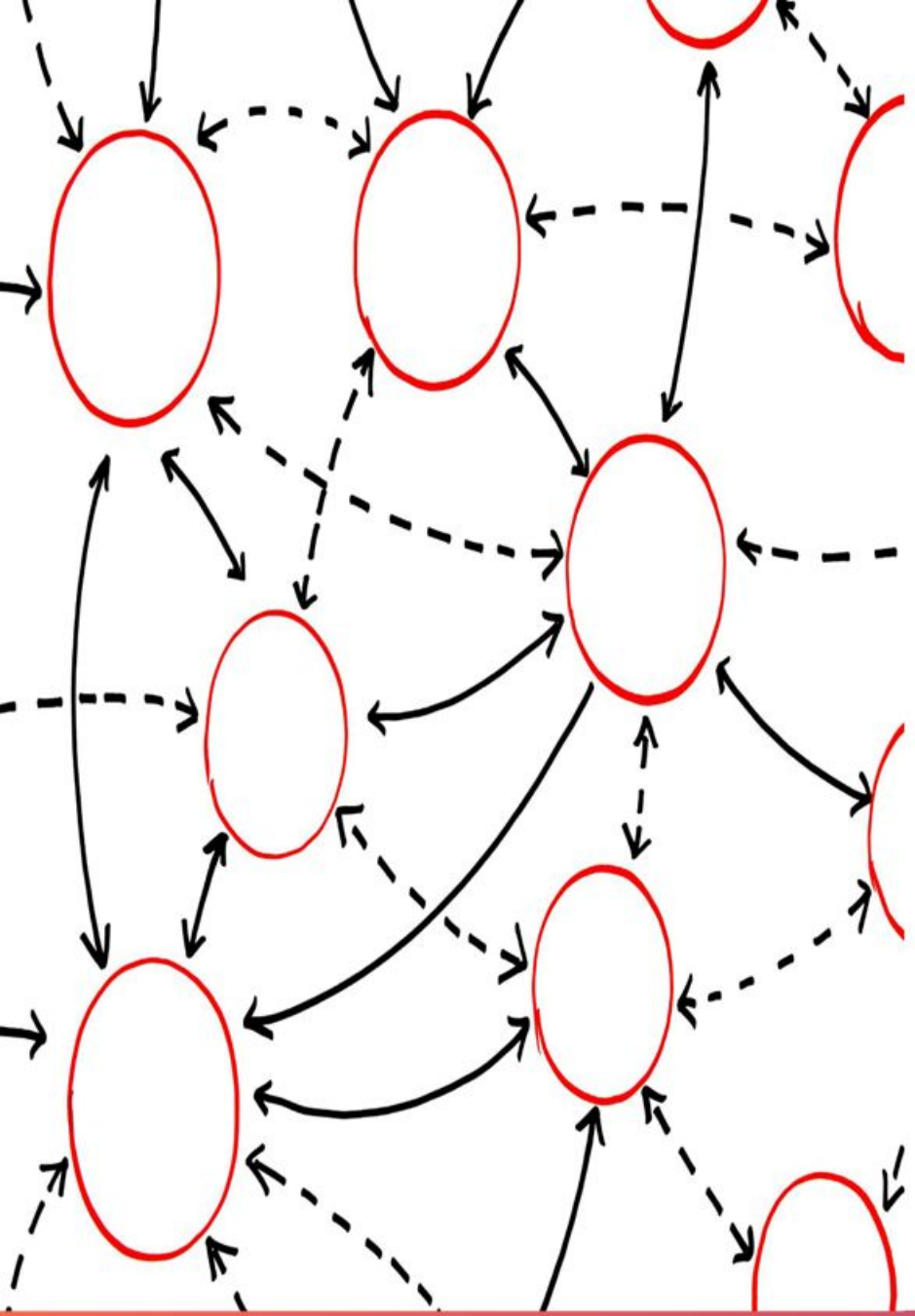
A child was born on a Delta Airways flight from New York to Mumbai. The plane was registered to the US. The child was born to an Indian Father and a US Mother. Upon landing the child was denied entry into India as it did not have a US Passport or a Visa to visit India. Were Indian authorities correct?

Model	Core content qualities	Reasoning and structure	Use of authorities and citations	Linguistic quality and usability	Safety, ethics, and compliance	Total Score
Staram Agent	5	5	5	5	5	25
Staram Agent1	5	5	5	5	5	25
ChatGPT Free	1	3	3	4	4	15
Claude Sonnet 4.5	3	4	5	4	4	20
Gemini 3	1	3	1	4	3	12
GPT 5.2	4	4	5	4	4	21

Test Query 2 -

A Power Of Attorney was Affirmed before the Indian Embassy in Washington. The Attorney holder uses it to dispose of property in India. Is that valid?

Model	Core content qualities	Reasoning and structure	Use of authorities and citations	Linguistic quality and usability	Safety, ethics, and compliance	Total Score
Staram Agent	5	5	5	5	5	**25**
Staram Agent1	5	5	5	5	5	**25**
ChatGPT	5	5	3	4	4	**21**
GPT 5.2	5	4	4	4	4	**21**
Gemini 3	5	4	1	4	4	**18**
Claude Sonnet 4.5	1	3	5	5	4	**18**



STARAM

Frameworks we are studying...

- RAP Framework

RAP uses planning and Monte Carlo Tree Search to create reasoning trees for complex legal query decomposition and evaluation.

- ReAct Framework

ReAct implements thought-action-observation loops, integrating reasoning with knowledge graph retrieval for modular legal responses.

- Chain of Thought Framework

Chain of Thought enables stepwise verbal reasoning by linking intermediate steps through knowledge graph entities for explainable synthesis.

Other Advanced Reasoning Frameworks for Legal Knowledge Graphs

- **Tree of Thought (ToT)** uses branching and pruning for promising legal reasoning paths.
- **Graph of Thought (GoT)** treats the knowledge graph as the medium of reasoning with traceability.
- **Self Consistency and Structured Debate** improve stability by sampling and merging diverse arguments.
- **Constitutional reasoning** encodes norms and guardrails ensuring consistent, compliant outputs.
- **Legal Grounding Controls** enforce validity, citation governance, and handle conflicting authorities.

CHALLENGES OF GRAPH DATABASES

- Graph complexity escalates rapidly as the dataset expands
- Maintaining accuracy and consistency demands significant effort
- Integration into common development stacks remains difficult
- High maintenance costs arise as graphs scale extensively
- Entity extraction and relation identification are prone to errors
- Increasing hops leads to exponential growth in complexity