Query-Aware Flow Diffusion for Graph-Based RAG with Retrieval Guarantees

Zhuoping Zhou* Davoud Ataee Tarzanagh* Sima Didari* Wenjun Hu Baruch Gutow Oxana Verkholyak Masoud Faraki Heng Hao
Hankyu Moon Seungjai Min

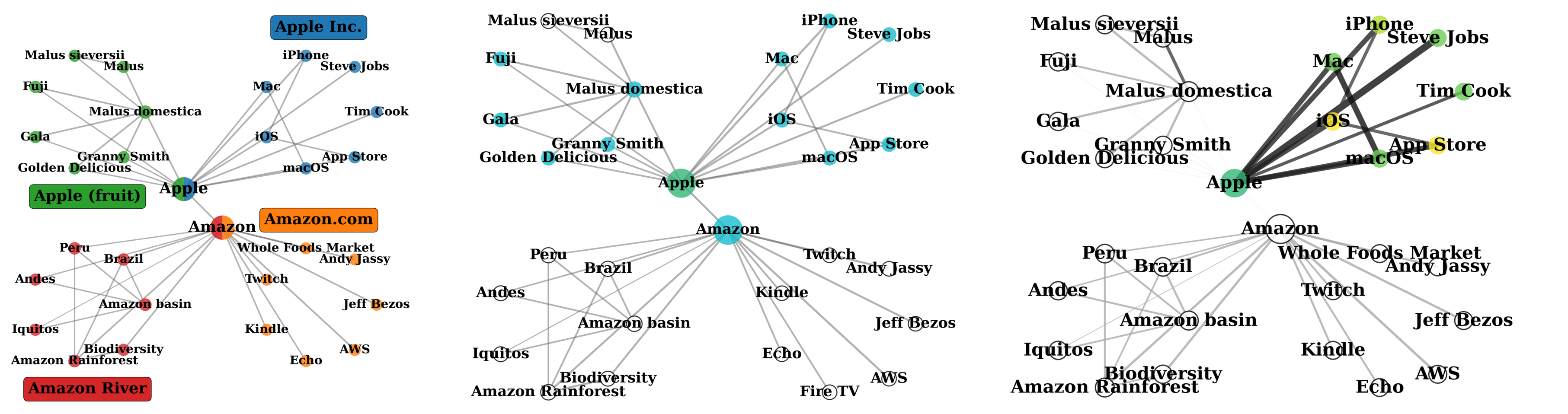
Samsung SDS Research America, Mountain View, CA, USA *Equal contribution

SAMSUNG SDS

Page: `qafd-rag.github.io`

Motivation & Contributions

Graph-based RAG captures multi-hop relationships that flat retrieval misses, but existing methods rely on heuristic subgraph selection with no guarantees.



Query: "Introduce Steve Jobs's products in Apple."

GraphRAG LightRAG QAFD-RAG

Our approach: Unlike **GraphRAG** (static communities) or **LightRAG** (1-hop ego-nets), **Query-Aware Flow Diffusion** reweights edges by query alignment so mass flows along semantically relevant paths only — with provable recovery guarantees.

Method

1. Find seeds. Extract query keywords via LLM; pick the top-N graph nodes whose embeddings match — these are the flow entry points.

2. Reweight edges. Each edge (u,v) gets a score that combines how well u,v are structurally connected with how strongly both align with the query:

$$\bar{w}(q,u,v) = \underbrace{H_{\mathrm{sim}}(h(u),h(v))}_{\text{structural (fixed)}} \cdot \left[1 + \tfrac{1}{4}\underbrace{\left(H_{\mathrm{sim}}(h(u),h(q)) + H_{\mathrm{sim}}(h(v),h(q))\right)}_{\text{query alignment (adapts per } q)}\right]$$

3. Diffuse mass. Inject mass at seeds; node scores $\mathbf{x}$ solve a convex quadratic whose curvature is set by the query-aware edges, and a push–relabel loop [5, 6] converges to the optimum while only touching nodes with excess mass — the solver is *lazy* and *local*.

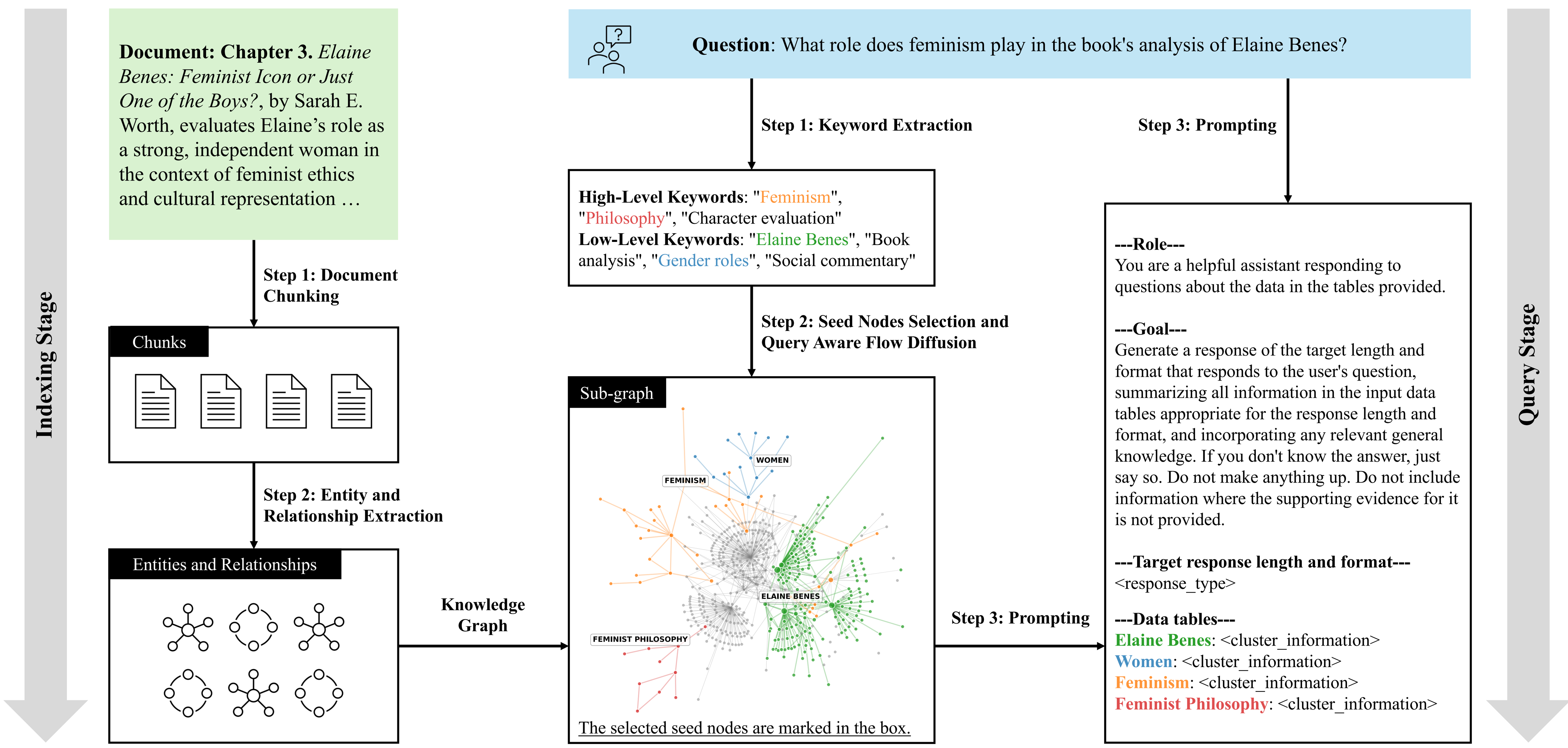
$$\min_{\mathbf{x}\geq 0} \ \tfrac{1}{2}\mathbf{x}^\top \mathbf{L}(q)\,\mathbf{x} \ + \ \mathbf{x}^\top(\mathbf{T}-\boldsymbol{\Delta}), \qquad \mathbf{L}(q) = \mathbf{B}\,\mathbf{W}(q)\,\mathbf{B}^\top$$

the query-aware weights $\mathbf{W}(q)$ enter through the Laplacian $\mathbf{L}(q)$ — they *shape* the objective, steering mass along query-aligned paths.

4. Extract subgraph. Nodes that end up holding mass $(\mathbf{x} > 0)$ form the retrieved context G_q passed to the generator.

Push–relabel (step 3):
1. inject $m_s = \alpha \sum_v T_v$ at each seed s; **2.** pick node v with $m_v > T_v$; **3.** compute $\bar{w}(q,v,\cdot)$ on-demand for neighbors; **4.** push the excess, update $\mathbf{x}$; **5.** repeat until $\sum_v \max(0, m_v - T_v) \leq \varepsilon$.

Framework Overview



Theoretical & Computational Properties

(1) Fast Convergence (Theorem 1)

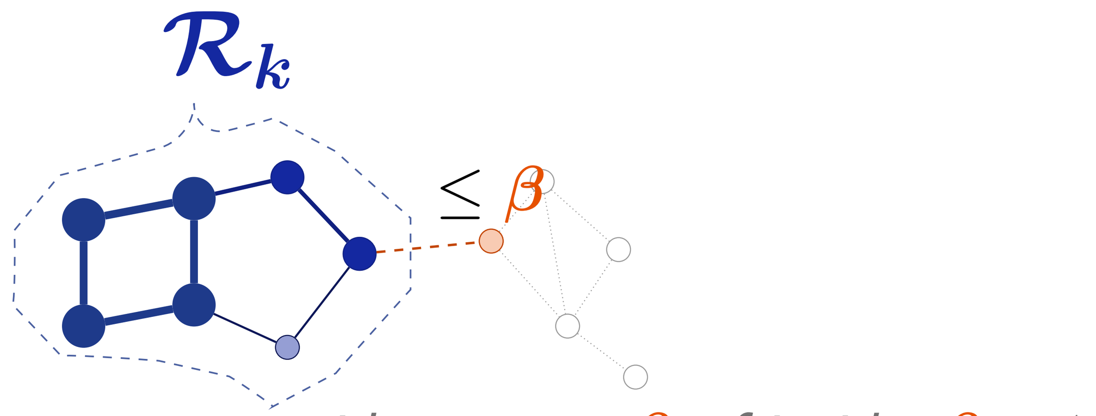
Reaches an ε-accurate solution in $\tilde{\mathcal{O}}(\bar{d}\,\|\boldsymbol{\Delta}\|_1 \log(1/\varepsilon))$ iterations — runtime grows with the **explored subgraph**, not the **full knowledge graph**.

Parameters: ε = target accuracy; $\bar{d}$ = max degree of visited nodes; $\|\boldsymbol{\Delta}\|_1$ = total seed mass at query entry points.

(2) Correct Recovery (Theorem 2)

Let $\mathcal{R}_k$ denote the set of query-relevant nodes (those that should be retrieved for query q_k). Under mild signal-to-noise conditions on node embeddings:

- **Completeness:** every node in $\mathcal{R}_k$ receives positive flow.
- **Bounded leakage:** mass outside $\mathcal{R}_k \leq \beta \cdot$ (mass inside $\mathcal{R}_k$).

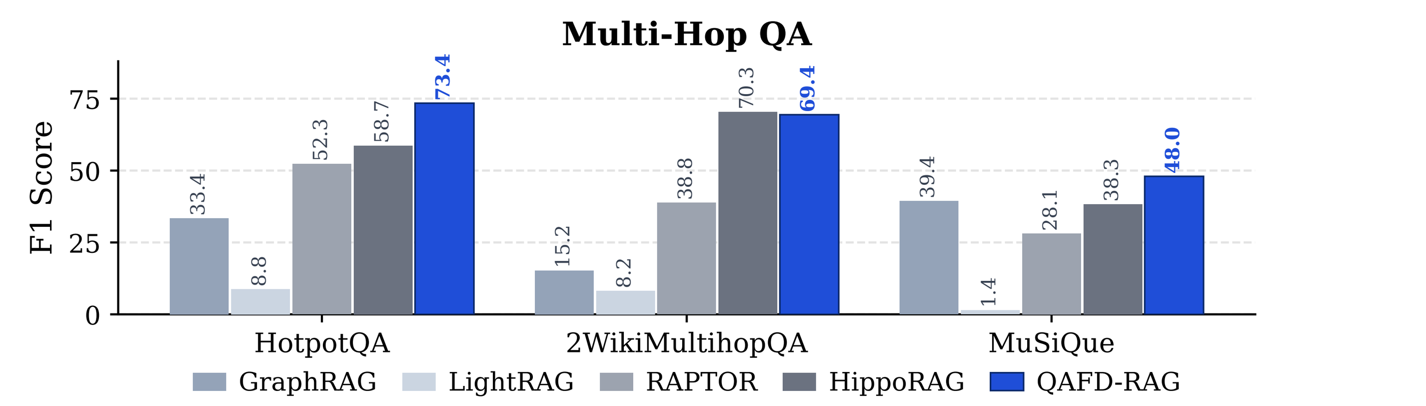


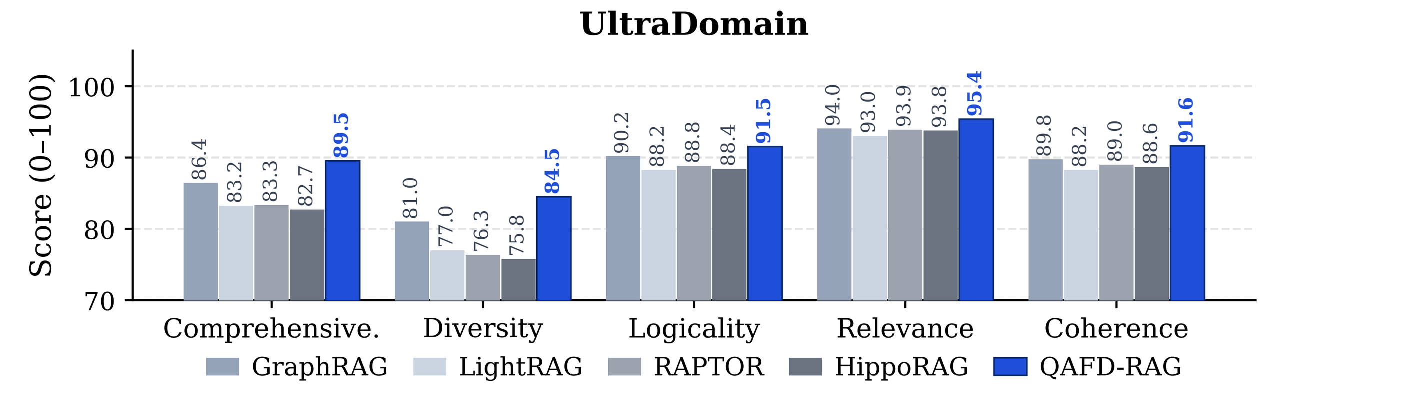
mass outside stays $\leq \beta$ of inside, $\beta \ll 1$

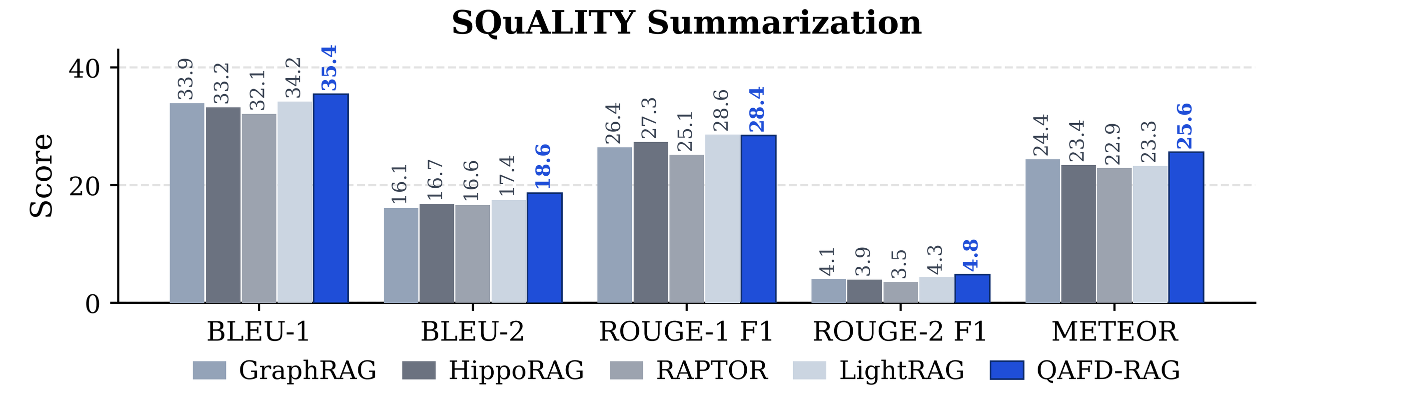
(3) Computational — Scales & Lazy

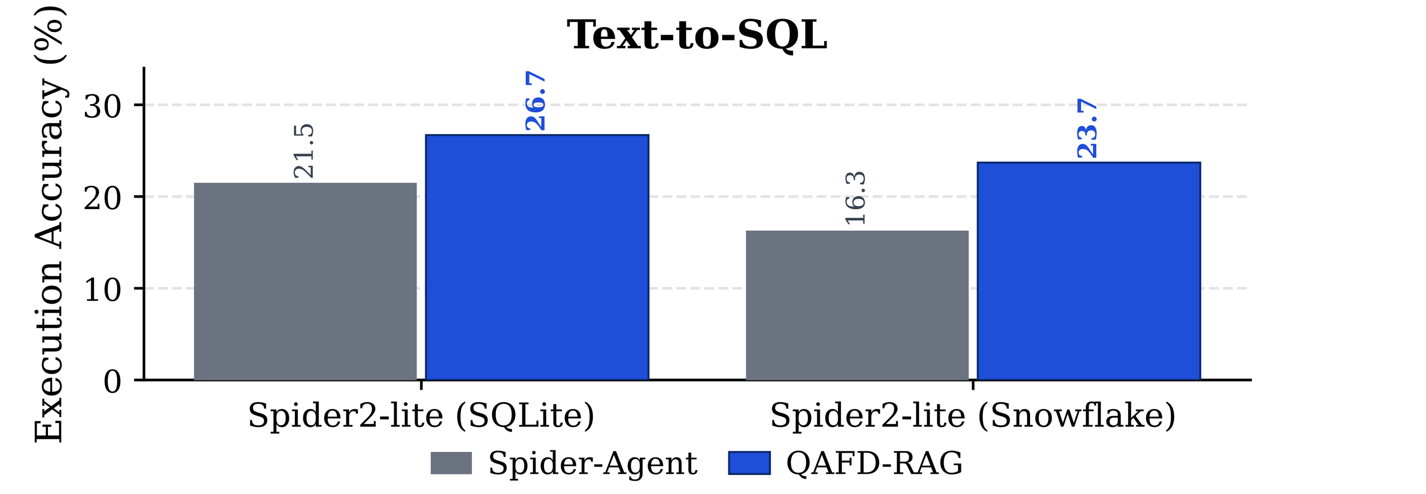
- **Scales to large graphs:** only the needed portion is touched — **no full scans**.
- **Lazy edges:** weights are computed **on-demand** as flow propagates.

Experimental Results









References

[1] Edge et al. *From Local to Global: A Graph-RAG Approach.* 2024.
[2] Guo et al. *LightRAG: Simple and Fast Retrieval-Augmented Generation.* arXiv:2410.05779, 2024.
[3] Jiménez-Gutiérrez et al. *HippoRAG: Neurobiologically-Inspired Long-Term Memory for LLMs.* NeurIPS 2024.
[4] Sarthi et al. *RAPTOR: Recursive Abstractive Processing for Tree-Organized Retrieval.* ICLR 2024.
[5] Fountoulakis et al. *Flow-based algorithms for improving clusters.* JMLR 2023.
[6] Wang, Fountoulakis, Henzinger, Mahoney, Rao. *Capacity Releasing Diffusion for Speed and Locality.* ICML 2017.
[7] Lei et al. *Spider 2.0: Evaluating LMs on Real-World Enterprise Text-to-SQL.* ICLR 2024.