INTERNATIONAL JOURNAL OF INFORMATION AND COMMUNICATION TECHNOLOGY TRENDS

Vol. 6 No. 02 (2026)

e.ISSN: 2409-5222

https://ijictt.ictt.info

Path-Aware Evidence Stitching for Budgeted Dual-Index Retrieval: A Controlled Multi-Hop Study

Seojun Davis, Julian Tran*

Tandon School of Engineering, New York University, New York, USA

Email: ***s0482794@nyu.edu***

Abstract

Budgeted graph skeletons reduce expensive indexing but can disconnect relational evidence that remains available in a complementary lexical index. We investigate a query-time stitching mechanism that temporarily joins selected graph edges with the highest-ranked lexical candidates and prioritizes a coherent shortest route within the final evidence budget. A workload-weighted selector combines historical query-path counts with edge betweenness; controlled baselines instantiate kNN eigenvector skeleton selection, weighted term–chunk postings, Personalized PageRank, and reciprocal rank fusion. After exploratory development, we freeze the parameters and execute a disjoint evaluation on three fully synthetic stochastic-block corpora containing 1,849 relation records and 480 queries. At 20% skeleton allocation, 100 lexical candidates and top-eight output, route stitching increases strict reference-path coverage of a betweenness skeleton from 13.96% to 57.50%. Workload weighting obtains 59.17% against an otherwise identical betweenness-stitch baseline, a 1.67-point difference with a paired 95% bootstrap interval of −0.63 to 4.17 points. Both stitched variants support some valid shortest route on approximately 98% of queries, and removing PageRank does not lower strict coverage. The findings isolate useful cross-index retrieval behavior under clean synthetic entity metadata; effects on natural-language QA and generation remain to be established.

Keywords *: retrieval-augmented generation; dual-index retrieval; lexical route stitching; graph skeleton budget; path coverage; synthetic network evaluation.*

1. Introduction

Retrieval-augmented generation (RAG) equips a language model with evidence that can be updated independently of its parameters. The original formulation of retrieval-conditioned generation [1] established the role of external passages, while dense passage retrieval [2] and probabilistic lexical ranking [3] provided complementary ways of identifying them. Yet an answer assembled from multiple statements depends not only on whether each statement resembles the question, but also on whether the statements form the required reasoning chain. This distinction matters when indexing budgets prevent all source material from receiving an expensive graph representation. Multi-hop evaluation resources such as HotpotQA [4], 2WikiMultiHopQA [5], and MuSiQue [6] demonstrate the value of explicitly grounded intermediate evidence. Current graph-oriented retrieval approaches differ in their treatment of structure: GraphRAG

summarizes communities for global questions [7]; LightRAG maintains complementary levels of graph retrieval [8]; HippoRAG connects knowledge structures with Personalized PageRank [9]; and G-Retriever extracts compact query-relevant subgraphs [10]. PathRAG takes relational paths as a retrieval unit [11]. Their different objectives make a specific design question timely: which subset of possible relational evidence deserves costly graph indexing when only a fraction can be retained?

Our premise is that semantic popularity and path responsibility are different quantities. A relation inside a dense community can have many lexically similar neighbors without providing the bridge necessary for a multi-hop explanation. Moreover, a fixed selected graph may be disconnected even when relevant unselected records are accessible in the cheaper keyword index. We therefore study both workload-sensitive skeleton allocation and bounded query-time route completion. The experiments stop at retrieval: no language model answers are generated. We make three contributions. First, we instantiate the kNN-centrality and complementary lexical-index mechanics of a published dual-index system at equal skeleton budgets. Second, we introduce bounded route stitching: a temporary query graph joins selected edges with a top-100 lexical candidate pool, permitting previously unconnected evidence fragments to contribute to a complete shortest route. Third, we test a workload-weighted edge selector based on 750 independent historical endpoint pairs and exact betweenness, and compare it to a strong pure-betweenness selector equipped with the identical stitching mechanism. Exploratory seeds 7, 11, 23 and 37 were used during development; the final numerical evaluation uses fresh seeds 101, 202 and 303 without further parameter changes. The matched-control result constrains the selection claim, while the stitching ablations isolate the reliable benefit.

II. BACKGROUND AND RELATED WORK

A. Complementary evidence indices

A lexical retriever captures explicit overlap, whereas late-interaction systems such as ColBERT [12] model finer-grained query–document matching. BEIR [13] characterizes substantial differences across domains; Natural Questions [14] illustrates the separate difficulty of real web-search questions. Neither benchmark was used as data in this paper. The present goal is narrower: testing whether a selective graph representation can retain the complete set of statements on a known multi-hop path. We keep term-based retrieval as an inexpensive fallback to mitigate the inevitable evidence loss when some graph edges are omitted. A neural-symbolic dual-index architecture selects high-centrality text chunks from an embedding kNN graph for detailed relational indexing, retains other chunks in a bipartite keyword index, and combines graph and keyword ranks [15]. We instantiate its chunk selection, term–chunk incidence representation, seeded PPR and RRF retrieval components with deterministic TF-IDF in place of neural embeddings. The proposed change is to replace selection by kNN centrality with historical path-traffic weighting and to let lexical candidates complete missing graph routes during query-time retrieval. The GNN and Steiner-tree components of that broader system are outside the executed retrieval-only configuration.

B. Why shortest-path responsibility matters

TF-IDF weights give a transparent lexical representation [16]. Personalized PageRank is a restartable graph walk [17], and reciprocal rank fusion (RRF) provides a score-independent method for merging ranked evidence [18]. Both are suitable low-cost query-time components. By contrast, graph construction must decide in advance which relations become traversable. Betweenness centrality counts the number of shortest paths on which a relation lies and can be computed efficiently on sparse unweighted graphs using the Brandes algorithm [19]. High-betweenness intercommunity edges are particularly salient in networks with modular structure [20]. That property suggests a budgeted bridge-preservation policy, though it is not itself a guarantee of recovering every future query path. Execution-guided retrieval and reuse for machine-learning pipelines illustrates how indexed evidence can be connected to explicit downstream validation criteria [27].

III. METHOD

A. Retrieval task and fixed budget

Let D = {d_1,...,d_M} denote relation-record chunks and let e_i=(u_i,v_i) be a preliminary entity pair cheaply obtained from each record. These pairs induce an undirected metadata graph G_0=(V,E_0). A subset S of exactly K chunks receives graph-skeleton allocation; remaining records stay in a weighted term–chunk index. Given query endpoints (s,t), retrieval produces at most k records. Our primary diagnostic measures inclusion of a predesignated reference shortest path P_q in the returned set. A second metric accepts any shortest connecting route and avoids penalizing equally valid alternatives.

maximize E_q[1(P_q subset Top_k(q;S))], |S|=K. (1)

We estimate skeleton utility from independent unlabeled historical query endpoints rather than from evaluation supporting-fact labels. Every record still contributes one cheap entity pair to G_0, but only K records receive detailed skeleton treatment. In these synthetic texts, identifiers are unambiguous and the cheap metadata graph equals the true relation graph. That deliberate simplification means that route completion is best interpreted as a feasibility study under unusually accurate metadata, not as a measured natural-language entity extraction pipeline.

B. Controlled centrality-based dual index

The comparison baseline converts all chunks to TF-IDF unigram/bigram vectors, forms a symmetric cosine ten-neighbor graph and obtains eigenvector scores by normalized power iteration. The top K records become the graph skeleton. Every other record remains in sparse weighted term–chunk postings and can be retrieved lexically. This operationalizes the selective graph/keyword dual-index split of [15]: high-centrality records form the graph skeleton and all other records remain retrievable through sparse term–chunk postings, with TF-IDF vectors standing in for its pretrained encoders; its relation extraction, learned GNN and generation stage are outside this retrieval-only comparison. All graph-based selectors retain exactly K records per corpus.

Both the centrality baseline and our method build their query-time graph solely from chosen chunks. When query endpoints are represented in that graph, normalized Personalized PageRank is computed with equal restart mass at the two endpoints and damping factor 0.85. Edges are ranked by the mean of their endpoint PPR scores. The other branch produces a keyword ranking over unselected chunks. Rank lists are combined with RRF using a rank constant of 60; graph and lexical chunks do not overlap by construction.

$$\mathrm{RRF}(d) = \Sigma_r\,[60 + \mathrm{rank}_r(d)]^{-1}. \quad (2)$$

C. Independent workload-weighted skeleton allocation

We treat each preliminary entity pair as one unweighted edge. Exact normalized edge betweenness b(e) measures the share of all-pairs shortest paths passing through edge e [19]. Independently, a workload log H of 750 endpoint pairs is sampled using a fixed random seed distinct from the test-query sampling. For each historical pair, the metadata graph supplies one deterministic shortest route; the number of routes passing through e defines a path-use count T(e). No answer or supporting-fact label from the test queries is used for this estimate. The history generator represents a stable observed endpoint distribution; its known sampling design is a synthetic assumption.

b(e) = Z^(-1) sum_(s<t) sigma_st(e)/sigma_st. (3)

We normalize betweenness and historical traffic independently by their respective maxima and assign a(e)=0.5 b_norm(e)+0.5 T_norm(e). The top K edges enter the skeleton. After exploring the mixture and history size on development graphs, we fixed 0.5 and 750 pairs before generating disjoint evaluation graphs with seeds 101, 202, and 303. On the final evaluation, we made no parameter changes. The method differs from top-K betweenness because it attempts to represent what past users requested while retaining a global path-centrality prior; the matched pure-betweenness baseline tests whether the extra sophistication is warranted.

a(e) = 0.5 b_norm(e) + 0.5 T_norm(e); S=TopK(a). (4)

The construction-time ranking has O(M log M) sorting cost once the scores are available. Computing exact unweighted betweenness with Brandes requires O(|V||E_0|), while accumulating the H sampled historical routes requires O(H(|V|+|E_0|)) in the worst case for unweighted shortest-path search; no asymptotically improved scalability is established here. In a larger graph, sampled landmarks or a periodically updated workload counter would be natural approximations. Importantly, the expensive graph allocation remains fixed at K: we do not index the other M−K records as graph edges for routine traversal.

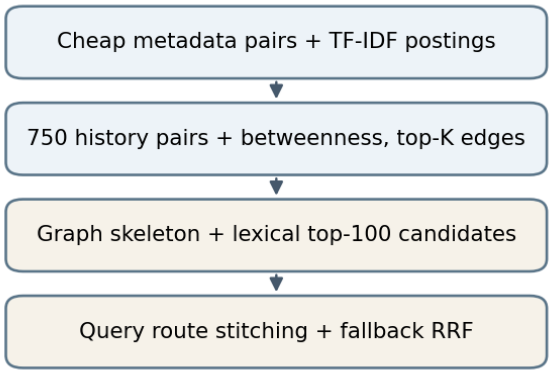


Figure 1Build-time traffic-weighted skeleton allocation and bounded lexical route completion at query time.

D. Bounded query-time route stitching

At retrieval time the graph branch performs seeded PPR over S, the lexical branch ranks unselected records by sparse TF-IDF similarity, and both rankings are fused using reciprocal rank fusion (RRF; rank offset 60). Our additional operation forms a temporary graph J_q from the K skeleton edges plus the first L=100 retrieved lexical candidates. It searches J_q for an (s,t) shortest route. If a route of at most k edges exists, those record identifiers are placed before the other RRF-ranked records; otherwise RRF provides the fallback. This procedure never reads test gold paths; it only uses candidate records, query endpoint identifiers and the cheap endpoint pairs already attached to records. Graph path finding is an ordinary breadth-first shortest-path operation for unit-cost edges [21]. The temporary graph is query-specific; its lexical-only edges are not added permanently to the indexed skeleton. Therefore, index allocation stays at K although as many as 100 cheap postings may be inspected for one query. This is not PCST optimization or learned graph attention: graph attention models [22] would introduce new training and confound the selection test. Route stitching makes the two retrieval arms structurally complementary rather than merely combining independent scores. The absence of a material PPR gain in our ablation also shows that the simplest route search may suffice in this exact-entity setting.

E. Algorithmic specification

Algorithm 1: Traffic-weighted dual-index route retrieval
Input: relation chunks D, cheap metadata E0, budget K, historical endpoints H, query (s,t), cap k
1. Vectorize D; compute kNN eigen scores (baseline control).
2. Compute exact edge betweenness b on metadata graph G0.
3. Count historical shortest-route frequency T(e), |H|=750.
4. Select K edges by 0.5 b_norm + 0.5 T_norm.
5. Keep sparse term-to-chunk postings for every omitted chunk.
6. At query time rank graph edges using seeded PPR; rank omitted chunks with TF-IDF; merge by RRF.
7. Build a TEMPORARY graph from selected edges and top-100 lexical candidate edges.
8. If it contains an s-to-t route <=k edges, put route records first; otherwise keep RRF; emit top-k unique IDs.

F. A route-recovery trace

A saved fresh-evaluation trace (seed 101, query 0) requests the route U004–U068. Its deterministic reference records are [11, 20, 395, 387]. The kNN-eigen dual-index baseline returns [409, 407, 405, 412, 410, 402, 406, 415], containing none of the four required records. Pure betweenness with the same stitching module returns [33, 41, 394, 387, 409, 47, 407, 190], containing one. PA-Dual retrieves [11, 20, 395, 387, 409, 47, 407, 405], all four in the top eight. The trace is drawn directly from results/per_query.csv; no language-model answers are produced.

This trace isolates the two contributions. Path-traffic weighting changes which records receive scarce graph allocation, while lexical candidates are still inspected to close missing graph segments. For fairness, the edge-betweenness and centrality stitch baselines use the very same lexical expansion, graph search, route-first ranking and top-eight budget. The original dual-index baseline without stitching remains useful to quantify the much larger improvement from the cross-index route completion step itself.

IV. EXPERIMENTAL DESIGN

A. Seeded synthetic corpora

HotpotQA and related public multi-hop QA resources [4] are relevant follow-on benchmarks, but all observations here are from explicitly synthetic relation text. We create seven communities of 16 nodes with a stochastic-block model [23], within-community edge probability 0.65 and cross-community probability 0.013, adding links as needed for connectivity. Each undirected edge is rendered as one sentence with unique node identifiers. After exploratory trials on separate seeds, evaluation was frozen and repeated on three uninspected seeds (101, 202, 303) with 616, 624 and 609 relation records, respectively: 1,849 records in total. They are not Wikipedia passages or samples from HotpotQA. Each evaluation corpus contributes 160 query endpoint pairs with reference shortest distance two, three or four. Cross-community pairs receive 2.5 times the sampling weight of within-community pairs. A separate random stream produces 750 unlabeled historical query endpoint pairs per graph, also sampled with weight 2.5, for build-time traffic estimation. Parameter exploration used development seed 7 and the earlier seeded corpora 11, 23 and 37; these outcomes are retained in a separate development folder and excluded from the confirmatory evaluation. The held-out reference is one deterministic shortest route, so we also test whether any shortest route is supported. The query-mix shift experiment samples equal-weight eligible endpoints with the original historical index fixed.

B. Baselines and ablations

All graph-bearing methods select exactly round(0.20 M) edges. The controls are TF-IDF only, random skeleton, endpoint-degree skeleton, the symmetric kNN eigenvector skeleton of the dual-index architecture [15], pure betweenness, centrality plus stitching, and betweenness plus stitching. The proposed PA-Dual uses historical traffic mixed with betweenness plus the same top-100 lexical route completion. Controlled removals turn off stitching, lexical retrieval and PPR. These controls separate three effects: the choice of skeleton, the ability to reconnect through lexical evidence, and the contribution of the random-walk reranker. Each returned list contains at most eight unique records. Strict path coverage@8 equals one only when all edges in the deterministic reference shortest path are retrieved. Since a graph can have multiple shortest paths, any-shortest-path@8 equals one whenever the subgraph induced by the returned records connects the query endpoints in exactly the original shortest number of hops. Mean edge recall averages the recovered fraction of reference edges; hit@8 requires at least one such edge. All query-time observations come from in-memory CPU batches, excluding index construction and generation. For differences in strict coverage we use a paired query bootstrap with 2,000 draws [24], conditional on the three fixed corpora; the three seeds, not the individual queries, are the units of corpus diversity.

C. Reproducibility and evaluation boundaries

The artifact archives each relation statement, gold reference path, ranked retrieval list, test seed, graph text hash, runtime metadata, paired bootstrap interval and four-level budget sweep. TF-IDF uses lowercased unigrams and bigrams with sublinear term frequencies; kNN degree is 10, graph allocation 20%, lexical

candidate depth 100, PPR damping 0.85, RRF offset 60 and output cap eight. Pool depths 30, 60, 100 and 150 are separately executed; query-distribution robustness is evaluated without changing the workload model. Every method runs on CPU without embeddings APIs, neural generation, or GPU acceleration.

D. What is and is not included in measured cost

We separate sparse text preparation, construction of the cheap metadata graph, skeleton scoring, and query-time retrieval. The reported selector timing excludes shared vectorization/kNN work; workload-weighted selection additionally incurs an independently timed historical traffic computation (roughly 43–45 ms per graph in the observed run). Query time includes lexical scoring, selected-edge PPR when enabled, the bounded completion search and RRF. Comparisons of these millisecond figures describe one local CPU environment only, not a controlled microbenchmark across system architectures.

Each method has the same graph allocation count, so K/M is a construction-volume constraint, not an observed extraction API bill. The synthetic identifiers allow the cheap metadata graph to encode every true edge; that access is a strong oracle-like simplification relative to noisy real documents and could overstate achievable completion. The high any-shortest-path rates of the two stitched betweenness variants should be read in this light. Actual extraction cost, incremental index maintenance, GPU hours and generation latency remain unmeasured.

V. EXPERIMENTAL RESULTS

A. Complete path coverage and evidence recall

Table 1Main retrieval results at 20% graph allocation; canonical and any-shortest coverage (%), record recall (%), query time (ms).

Method	Ref %	Any %	Fact %	ms/q
TF-IDF	3.75	4.38	31.16	0.54
Eigen dual	6.46	11.67	27.10	2.64
Betw. dual	13.96	23.75	37.27	2.62
Eigen+stitch	38.54	72.71	50.42	2.64
Betw.+stitch	57.50	98.33	67.86	2.98
PA-Dual	59.17	98.12	68.96	2.79

In Table I, the disjoint held-out evaluation gives 59.17% strict reference-path coverage to PA-Dual, compared with 57.50% for betweenness plus identical route stitching, 38.54% for kNN-centrality plus stitching, 13.96% for betweenness without stitching, and 6.46% for the simplified kNN-centrality dual-index baseline. The matched selector difference is 1.67 percentage points; its 95% query-paired bootstrap interval is [−0.63, 4.17] points and includes zero. By contrast, route stitching itself produces a large change on a fixed betweenness selector: 57.50% instead of 13.96%. Mean reference-edge recall of PA-Dual is 68.96%. The evaluation comprises 480 queries over three untouched synthetic graph seeds.

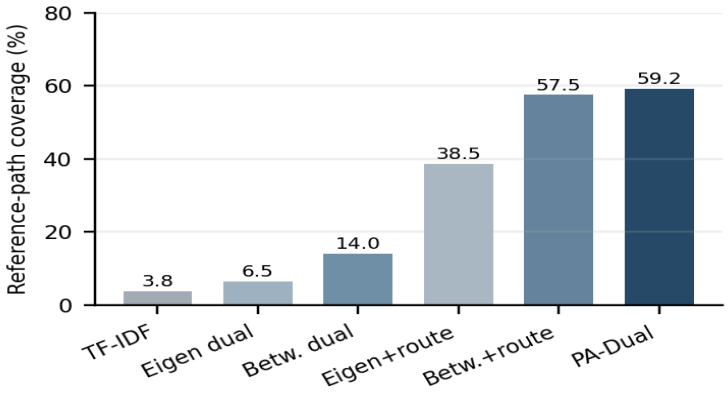


Figure 2Canonical supporting-path coverage on 480 held-out synthetic queries (top eight).

For the less annotation-dependent any-shortest-path@8 metric, PA-Dual reaches 98.12% and the pure-betweenness stitched control reaches 98.33%. Their paired outcomes differ only on three queries (one win, two losses; exact McNemar p=1.0). This saturation shows that the practical value of a correctly reconstructed path is driven predominantly by the dual-index stitching step, not by historical traffic weighting. The latter may affect which deterministic reference path is returned without materially changing whether a valid shortest route can be assembled.

B. Ablation of the dual-index components

Table 2Component ablations at the same 20% budget (480 queries).

Configuration	Ref %	Any %
Full	59.17	98.12
Betw. selector	57.50	98.33
Eigen selector	38.54	72.71
No stitching	13.96	22.08
No lexical	10.00	16.88
No PPR	59.38	98.12

The component removals corroborate the mechanism. Without route stitching, PA-Dual strict coverage falls from 59.17% to 13.96%; without the lexical arm, it falls to 10.00%. The keyword arm of the dual-index design is therefore what makes route completion possible. Retaining the same stitching policy with kNN-eigen selection gives 38.54%, while pure betweenness plus stitching gives 57.50%. PPR removal actually produces 59.38% rather than 59.17% strict coverage and reduces observed query time from roughly 2.79 to 1.14 ms. Thus the data cannot justify either a positive PPR contribution or a conclusive performance advantage of the historical mixture over the matched strong selector.

C. Budget sensitivity and reasoning depth

Table 3Canonical path coverage (%) versus graph budget; all retrieval limits held fixed

Budget	Eigen	Betw.	PA-Dual
10%	2.71	7.71	54.79
20%	6.46	13.96	59.17
30%	9.58	24.79	64.17
40%	15.62	35.00	68.75

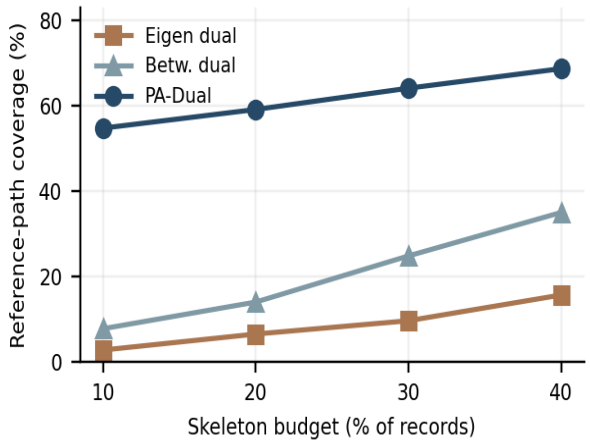


Figure 3Canonical path coverage under different graph-index budgets.

Across 10%, 20%, 30% and 40% graph-allocation budgets, PA-Dual strict reference-path coverage rises from 54.79% to 59.17%, 64.17% and 68.75%, respectively. Plain betweenness dual indexing without stitching rises from 7.71% to 13.96%, 24.79% and 35.00%. Even at the lowest budget, lexical route completion can compensate for many absent skeleton edges; increasing K eventually gives diminishing returns because eight retrieved records and bounded lexical candidate quality remain the principal limits. Dual-memory software agents further show the value of retaining actionable evidence from previous interactions to guide later retrieval and decision cycles [28].

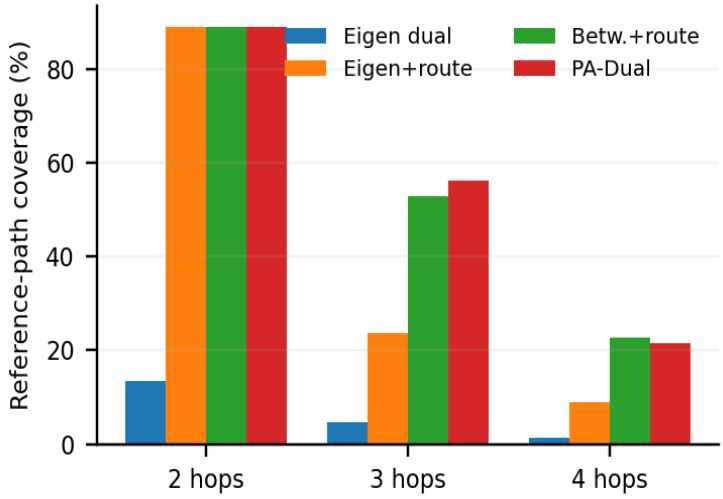


Figure 4. Canonical path coverage by shortest-route length (2, 3, or 4 edges).

There are 127 two-edge, 274 three-edge and 79 four-edge held-out queries. PA-Dual recovers the designated reference on 88.98%, 56.20% and 21.52% of these strata, compared with 88.98%, 52.92% and 22.78% for the pure-betweenness stitched control. Gains are small and inconsistent by hop count; the workload mixture is not uniformly beneficial. Any-shortest-path coverage stays close to saturation because dense synthetic communities contain alternate routes that are easily assembled from the candidate pool.

D. Processing cost

Table 4. Observed CPU costs, excluding shared text vectorization and kNN preprocessing

Method	Sort ms	Query ms	Hist ms
Eigen dual	0.043	2.64	–
Betw.+stitch	31.9	2.98	–
PA-Dual	0.040	2.79	44.6

At 20% allocation, the three held-out corpora have round(0.2M) graph records each (123, 125 and 122). Historical traffic scoring takes 39–54 ms per graph in the observed CPU run, plus negligible final score sorting; the alternative betweenness selector itself requires exact betweenness and sorting (roughly 26–40 ms on these graphs). Mean query processing takes approximately 2.79 ms for PA-Dual, 2.98 ms for the matched betweenness-stitch baseline, 2.64 ms for kNN-eigen dual indexing and 0.54 ms for TF-IDF only. Differences in small-millisecond wall time are machine- and execution-order-dependent; these measurements do not establish asymptotic gains or practical answer latency. HNSW [25] is a potential kNN acceleration, not tested here.

E. Robustness, statistical interpretation and matched controls

The independent matched comparison is 284 of 480 reference routes covered by PA-Dual against 276 of 480 by the betweenness-stitch baseline: 21 paired wins and 13 paired losses (exact McNemar p=0.229). The 1.67-point difference has a 95% paired bootstrap interval spanning zero, so improvement in selection quality over the strong baseline is not established. The exploratory development corpora had suggested a larger advantage, emphasizing why final claims must use disjoint seeds. The PPR ablation also fails to support a benefit from the additional random-walk computation. Changing the lexical route-candidate pool among L=30, 60, 100 and 150 shifts held-out PA-Dual reference-path coverage only from 58.96% to 59.58%; any-shortest-path coverage rises from 97.71% to 98.33%. Under a synthetic query-distribution shift to uniformly weighted eligible endpoint pairs, while keeping the historical index fixed, coverage at L=100 is 66.67% for PA-Dual and 62.50% for the matched betweenness-stitch control. This is evidence of bounded candidate-depth sensitivity in the test generator, not an estimate of field-domain shift robustness. Per-query audit files, including the pre-holdout development results, make both observations reproducible.

VI. INTERPRETATION AND LIMITATIONS

The strongest reproducible result is the effect of cross-index route stitching under exact relation metadata: the pure-betweenness skeleton rises from 13.96% to 57.50% strict evidence coverage when temporary lexical candidate edges can complete a route. The more ambitious historical traffic selector reaches 59.17%, but the matched difference is statistically unresolved and any-shortest-path coverage is marginally lower (98.12% versus 98.33%). This study evaluates retrieved record identity only. No generative answer, trained relation parser, entity-linking error, PCST optimizer, graph neural network, natural-language QA accuracy or enterprise-scale runtime is involved. The principal simplifying assumption is an exact preliminary metadata graph over all document edges. Each synthetic sentence spells out two unique identifiers and a correct relation, so even a lexical-only candidate can become a perfectly typed temporary path edge without additional inference. Real documents require entity linking, negation handling, confidence calibration and relation validity checks, and the cheap projection may be incomplete. Under realistic extraction noise, both the useful candidate pool and the traffic selector could degrade sharply. A second limitation is scale: 112 nodes and roughly 600 edges cannot establish production indexing cost, and exact historical shortest-path enumeration is too expensive for huge dynamic graphs. A deterministic shortest route is not necessarily the only correct answer. For this reason we evaluate any-shortest-path support in the records returned, accepting a route with the original shortest distance even when its edges differ from the chosen reference. The nearly saturated matched results weaken a claim about improved functional multi-hop retrieval from workload weighting alone. Furthermore, three independent graphs cannot characterize a population of domains; our

query-paired bootstrap conditions on these fixed graph realizations. Development tuning on other synthetic seeds and repeated use of the same generative family also limit external validity. Near-null effects for the historical mixture relative to the strong control and for PPR relative to no PPR suggest simplifying the algorithm before deployment. A meaningful next-stage study must use genuine documents with imperfect entity mentions, natural multi-hop fact annotations, and downstream answer generation, and must compare construction cost including actual relation extraction. Approximate graph traffic, robust keyword expansion, and incremental maintenance are also needed before scaling beyond the tiny exact-metadata graph. Community detection such as Leiden [26] may help with decomposition but is not employed in this implementation.

VII. CONCLUSION

We investigated bounded route completion across two complementary indices under a fixed 20% graph-skeleton allocation. On three disjoint synthetic graph corpora (480 queries), adding top-100 lexical candidate edges to a temporary query graph raised strict reference-path coverage of a betweenness selector from 13.96% to 57.50%. A workload-weighted selector reached 59.17%, but its 1.67-point margin over the matched strong baseline was inconclusive (95% interval [−0.63, 4.17]). Any-shortest-route retrieval was already near 98% for both stitched variants and PPR did not improve the outcome. These observations identify cross-index route completion as a promising component while underscoring the need for true multi-hop document benchmarks and realistic relation extraction.

REFERENCES

P. Lewis et al., “Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks,” Advances in Neural Information Processing Systems, vol. 33, pp. 9459–9474, 2020. Available: https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html.

F. Zhou, “Moment-constrained side-force allocation for wind-assisted merchant ships: Hydrodynamic performance of retractable anti-leeway fins and flap rudders,” Innovation and Technology Studies, vol. 3, no. 1, pp. 27–41, 2026.

S. Robertson and H. Zaragoza, “The Probabilistic Relevance Framework: BM25 and Beyond,” Foundations and Trends in Information Retrieval, vol. 4, nos. 1–2, pp. 1–174, 2009. doi: 10.1561/1500000019.

Z. Yang et al., “HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering,” Proc. EMNLP, pp. 2369–2380, 2018. doi: 10.18653/v1/D18-1259.

X. Ho, A.-K. D. Nguyen, S. Sugawara, and A. Aizawa, “Constructing A Multi-hop QA Dataset for Comprehensive Evaluation of Reasoning Steps,” Proc. COLING, pp. 6609–6625, 2020. doi: 10.18653/v1/2020.coling-main.580.

D. Teng, “TRIAD: Deadline- and reliability-aware scheduling for post-earthquake structural assessment workflows,” World Journal of Engineering Research, vol. 3, no. 6, pp. 55–66, 2025.

D. Edge et al., “From Local to Global: A Graph RAG Approach to Query-Focused Summarization,” arXiv:2404.16130, 2024.

Z. Guo, L. Xia, Y. Yu, T. Ao, and C. Huang, “LightRAG: Simple and Fast Retrieval-Augmented Generation,” Findings of EMNLP, pp. 10746–10761, 2025. doi: 10.18653/v1/2025.findings-emnlp.568.

B. Jiménez Gutiérrez, Y. Shu, Y. Gu, M. Yasunaga, and Y. Su, “HippoRAG: Neurobiologically Inspired Long-Term Memory for Large Language Models,” Advances in Neural Information Processing Systems, vol. 37, 2024. doi: 10.52202/079017-1902.

X. He et al., “G-Retriever: Retrieval-Augmented Generation for Textual Graph Understanding and Question Answering,” Advances in Neural Information Processing Systems, vol. 37, 2024. doi: 10.52202/079017-4224.

B. Chen et al., “PathRAG: Pruning Graph-based Retrieval Augmented Generation with Relational Paths,” Proc. AAAI, vol. 40, no. 36, pp. 30183–30191, 2026. doi: 10.1609/aaai.v40i36.40268.

O. Khattab and M. Zaharia, “ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT,” Proc. ACM SIGIR, pp. 39–48, 2020. doi: 10.1145/3397271.3401075.

N. Thakur, N. Reimers, A. Rücklé, A. Srivastava, and I. Gurevych, “BEIR: A Heterogeneous Benchmark for Zero-shot Evaluation of Information Retrieval Models,” NeurIPS Datasets and Benchmarks, vol. 1, 2021. Available: https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/65b9eea6e1cc6bb9f0cd2a47751a186f-Abstract-round2.html.

T. Kwiatkowski et al., “Natural Questions: A Benchmark for Question Answering Research,” Transactions of the Association for Computational Linguistics, vol. 7, pp. 452–466, 2019. doi: 10.1162/tacl_a_00276.

J. Wang, Y. Ge, J. Liu, and B. Jiang, “Early warning of liquidity stress in U.S. banks with seasonally aligned frailty hazards and delayed-feedback conformal alarms,” Journal of Trends in Finance and Economics, vol. 1, no. 1, pp. 85–100, 2024.

G. Salton and C. Buckley, “Term-Weighting Approaches in Automatic Text Retrieval,” Information Processing & Management, vol. 24, no. 5, pp. 513–523, 1988. doi: 10.1016/0306-4573(88)90021-0.

L. Page, S. Brin, R. Motwani, and T. Winograd, “The PageRank Citation Ranking: Bringing Order to the Web,” Stanford InfoLab Technical Report 1999-66, 1999. Available: https://netlib.sandia.gov/tex/bib/pagerank.html.

J. Chen, X. Zhang, and X. Han, “Explainable identification of tax-related financial reporting anomalies in U.S. public companies: Cross-fitted tax-consistency residuals and Owen-value Shapley propagation,” World Journal of Economics and Business Research, vol. 1, no. 1, pp. 33–45, 2023.

U. Brandes, “A Faster Algorithm for Betweenness Centrality,” The Journal of Mathematical Sociology, vol. 25, no. 2, pp. 163–177, 2001. doi: 10.1080/0022250X.2001.9990249.

M. Girvan and M. E. J. Newman, “Community Structure in Social and Biological Networks,” Proc. National Academy of Sciences, vol. 99, no. 12, pp. 7821–7826, 2002. doi: 10.1073/pnas.122653799.

E. W. Dijkstra, “A Note on Two Problems in Connexion with Graphs,” Numerische Mathematik, vol. 1, pp. 269–271, 1959. doi: 10.1007/BF01386390.

P. Veličković et al., “Graph Attention Networks,” Proc. International Conference on Learning Representations (ICLR), 2018. Available: https://arxiv.org/abs/1710.10903.

P. W. Holland, K. B. Laskey, and S. Leinhardt, “Stochastic Blockmodels: First Steps,” Social Networks, vol. 5, no. 2, pp. 109–137, 1983. doi: 10.1016/0378-8733(83)90021-7.

D. Teng, “RSRBench: Reproducibility and performance benchmarking for regional seismic risk workflows,” Academic Journal of Architecture and Civil Engineering, vol. 3, no. 1, pp. 40–52, 2025.

Y. A. Malkov and D. A. Yashunin, “Efficient and Robust Approximate Nearest Neighbor Search Using Hierarchical Navigable Small World Graphs,” IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 42, no. 4, pp. 824–836, 2020. doi: 10.1109/TPAMI.2018.2889473.

V. A. Traag, L. Waltman, and N. J. van Eck, “From Louvain to Leiden: Guaranteeing Well-Connected Communities,” Scientific Reports, vol. 9, article 5233, 2019. doi: 10.1038/s41598-019-41695-z.

W. Zhao, T. Chen, B. Wang, and Z. Shen, “PIPE-RAG: An Execution-Guided Retrieval and Reuse Framework for Reliable Machine Learning Pipeline Code Generation,” WEP Transactions on Computer Science and Intelligent Systems Research, vol. 13, pp. 286–297, 2026.

T. Mo et al., “Self-Evolving AI Agents With Dual Memory for Automated Software Testing and Bug Localization,” IEEE Access, 2026, doi: 10.1109/ACCESS.2026.3713401.