Semantica: Tree-Initialized Semantic Overlays for Decentralized Search

Petru Neague
Delft University of Technology
Delft, The Netherlands
p.m.neague@tudelft.nl

Quinten Stokkink
Delft University of Technology
Delft, The Netherlands
q.a.stokkink@tudelft.nl

Naman Goel
University of Oxford
Oxford, United Kingdom
naman.goel@cs.ox.ac.uk

Johan Pouwelse
Delft University of Technology
Delft, The Netherlands
j.a.pouwelse@tudelft.nl

Abstract

In decentralized search, no party has a global view of what documents exist across users, so finding relevant ones at scale is challenging. We present Semantica to address this problem. The novel idea in Semantica is a *centroid tree* that organizes entries representing the centroids of peers' document clusters (interests). Each peer inserts its centroids into the tree separately, then selects peers associated with nearby tree entries as its initial outgoing semantic connections, thereby initializing a semantic overlay (peer graph). Each peer keeps a fixed number of connections, divided among its interests, and may refine them through referrals. Search queries follow a greedy route in this graph, and merge the document rankings that the contacted peers return. We evaluate the proposed method on AOL query logs with archived pages, a PubMedQA-derived retrieval workload, and Stack Overflow duplicate questions. After contacting at most 100 users, Semantica returns 57.94%, 67.57%, and 83.52% of the documents with highest cosine similarity to the query, respectively, while the designated target appears in its first ten results for 29.65%, 75.90%, and 62.45% of queries.

Ablations show that centroid tree produces a semantic overlay whose advantage over random initialization persists across graph-refinement procedures. In a controlled comparison, the resulting graph also outperforms an adapted prior multi-interest overlay.

CCS Concepts

• **Information systems** → **Language models**; **Similarity measures**; • **Networks** → *Overlay and other logical network structures.*

Keywords

decentralized search, peer-to-peer systems

ACM Reference Format:
Petru Neague, Quinten Stokkink, Naman Goel, and Johan Pouwelse. 2027. Semantica: Tree-Initialized Semantic Overlays for Decentralized Search. In *Proceedings of The 20th ACM International Conference on Web Search and Data Mining*. ACM, New York, NY, USA, 11 pages.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
The 20th ACM International Conference on Web Search and Data Mining, Hong Kong
© 2027 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-x-xxxx-xxxx-x/YYYY/MM

1 Introduction

Centralized search is the dominant model of information access on the public Web [2]. But it does not fit every search setting. For example, it can not reach documents that are not open to external crawling, such as clinical, enterprise, and legal repositories held within separate organizations [25]. A similar problem resurfaces in retrieval-augmented generation [31, 33]. Moreover, in centralized search, significant influence over visibility and ranking rests with a single party, which raises additional concerns [10]. In this paper, we study search in decentralized networks where documents are held by peers and no peer or third party has a global index.

Search queries in decentralized networks are answered by forwarding them to remote peers and combining the search results they return. Semantic overlay networks (SONs) are peer-to-peer networks in which connections reflect similarities between the resources provided by different peers [4]. For example, in the Web of Things, a participant may search machine-readable 'Thing Descriptions' for a device with the required capabilities [3], IoT systems discover devices and services from descriptive metadata [36], and federated and distributed Retrieval-Augmented Generation retrieve across autonomous document sources or agents [31, 33]. Although the resources differ, these applications share the general problem of matching a request to resources held at remote peers.

Earlier semantic overlays organized document indices or peer descriptions through structured overlays, hierarchies, and small-world networks [14, 27, 29, 34]. The idea is that each document can be summarized by a coordinate, called a *centroid*, that has the shortest distance to all of its contained information, for example, the average of all embeddings of all words in a text file. Documents remain with their owners; a centroid summarizes the embeddings of the documents it represents.

Representing a peer with one centroid can be inadequate when it holds documents about several unrelated subjects: a single mean may lie between document clusters and describe none of them well. This motivated other work that represented a peer using multiple centroids of local documents and maintained routing entries for peers with similar centroids [20]. Multiple centroids preserve local interests, but overlay construction can still begin from random contacts that share none of a peer's interests. A peer can improve its

neighborhood by asking a current contact to suggest other peers and replacing a weaker contact with a better-matching suggestion, a process we call *referral*. Since referrals can only suggest peers known to current contacts, interests missed by random initialization may be difficult to recover. More recent retrieval methods assume that a peer graph already exists, select among independently searched documents, or train indexes over predefined (i.e., static) document centroids [9, 18, 31, 33].

We introduce `Semantica`, in which the novel idea is a *centroid tree* that constructs the semantic peer overlay from the peers' interest centroids, rather than starting from random contacts or assuming the graph already exists. To give referral a useful starting point, we treat each cluster of related documents at a peer as an interest and represent it by a centroid, a vector summarizing the document cluster's semantic content. A centroid tree places similar summaries near one another, allowing nearby entries to provide candidates for initial direct links among document-owning peers. As a design choice, we fix each peer's total outdegree and divide the available connections among its interests in proportion to their cluster sizes, though `Semantica` can support other strategies too. For overlay construction, peers exchange peer identifiers and their centroid vectors. During retrieval, queries follow the resulting peer connections, and contacted peers return ranked results.

In summary, the main contributions of the paper are:

- `Semantica`, a decentralized search system based on the novel idea of *centroid tree* that builds the peer graph from peer interest centroids, giving each peer well-matched initial contacts without relying on random initialization, so that different parts of its document set connect to the peers relevant to that part.
- An evaluation on observed real-world (AOL-IA) interactions and workloads constructed from public (PubMedQA and Stack Overflow) content. The evaluation compares `Semantica` with decentralized and centralized baselines, examines the effects of different design choices (representation, initialization, graph refinement, and routing), and shows significant improvement in results.

2 Related Work

The most closely related work spans three areas: semantic peer organization, distributed and federated retrieval, and approximate nearest-neighbor indexes. This section provides a broad overview of the related work in the area, while in Section 5.2, we further discuss the baselines that are relevant for empirical comparison.

Semantic Peer Organization. Crespo and Garcia-Molina [4] group peers according to shared subject matter and direct queries toward matching groups, while leaving the links and routing within each group unspecified. SSW [14] represents each peer using the centroid of its largest local document group and publishes references for documents outside the semantic region assigned to that peer. iCluster [20] represents several interests per peer, obtains initial outgoing neighbors by following randomly selected peer edges, and maintains both semantically close and distant outgoing neighbors. Its query forwarding may branch to several peers. Doulkeridis et al. [8] recursively combine summaries produced by individual peers into connected semantic overlays and use coordinating peers to select among them. T-Man [11] has peers exchange limited sets of peer descriptions and rank the received peers according to the network structure being constructed. Semantica instead uses a tree of peer-interest centroids to select initial outgoing neighbors. Each peer maintains a fixed outdegree, divided among its interests according to the number of local documents they represent, with at least one outgoing slot retained for every interest. Outgoing neighbors may later suggest replacements for these slots.

Distributed and Federated Retrieval. pSearch [27] divides a semantic vector space into regions assigned to peers, stores document references in the corresponding regions, and searches the region nearest to a query together with nearby regions. It therefore organizes references to documents rather than directly connecting similar document owners. EZSearch [29] clusters document owners and uses a hierarchy to direct queries toward groups likely to contain matching documents. SemanticPeer [34] assigns related terms describing documents and queries to corresponding peers through a distributed lookup structure. It is therefore closer to a semantic distributed hash table (DHT) than to a network connecting similar document owners. FeB4RAG [31] is a benchmark for federated retrieval-augmented generation. It treats separate document sets as search sources and supports testing methods that decide which sources to search and how to combine the ranked documents they return. DRAG [33] assumes that peer links already exist and directs queries using topics inferred by a language model together with records of peers that helped with earlier queries. Graph-Diffusion [9] also assumes an existing graph. It combines each peer's document vectors into one summary and passes information from these summaries across existing links to guide queries; it does not construct the links.

Approximate Nearest-Neighbor Indexes. NN-Descent [7] constructs an approximate nearest-neighbor graph over data objects by repeatedly considering the neighbors of current neighbors. But it is an index-construction method, not a peer-to-peer routing protocol. P-stable [5] locality-sensitive hashing (LSH) uses random projections to place nearby vectors in the same buckets with high probability; the original method is centralized and held in memory. NearBucket-LSH [13] adapts this approach to peer-to-peer search by distributing the buckets among peers and searching both the matching bucket and selected nearby buckets. Hierarchical navigable small-world graphs (HNSW) [16] arrange stored items in graph layers: sparse upper layers support longer moves, while the denser bottom layer identifies nearby items. Pyramid [6] distributes HNSW across workers and uses a coordinator to select which worker indexes should search a query and combine their results.

3 Problem Formulation

Let $\mathcal{U} = \{u_1, \ldots, u_N\}$ denote the set of N participating peers. Each peer u_i stores a nonempty local document collection $\mathcal{D}_i$. Every document x has a stable identifier $\iota(x)$, such as an application identifier or content hash, and a normalized embedding $\mathbf{D}(x)$. For a document $d_{ij} \in \mathcal{D}_i$, we write $\mathbf{D}_{ij} = \mathbf{D}(d_{ij})$. All peers independently apply the same fixed encoder, so their embeddings lie in a common vector space. Copies with the same identifier denote the same document, and every peer storing one is an owner. Identifiers are used for deduplication; they do not need to be collectively allocated

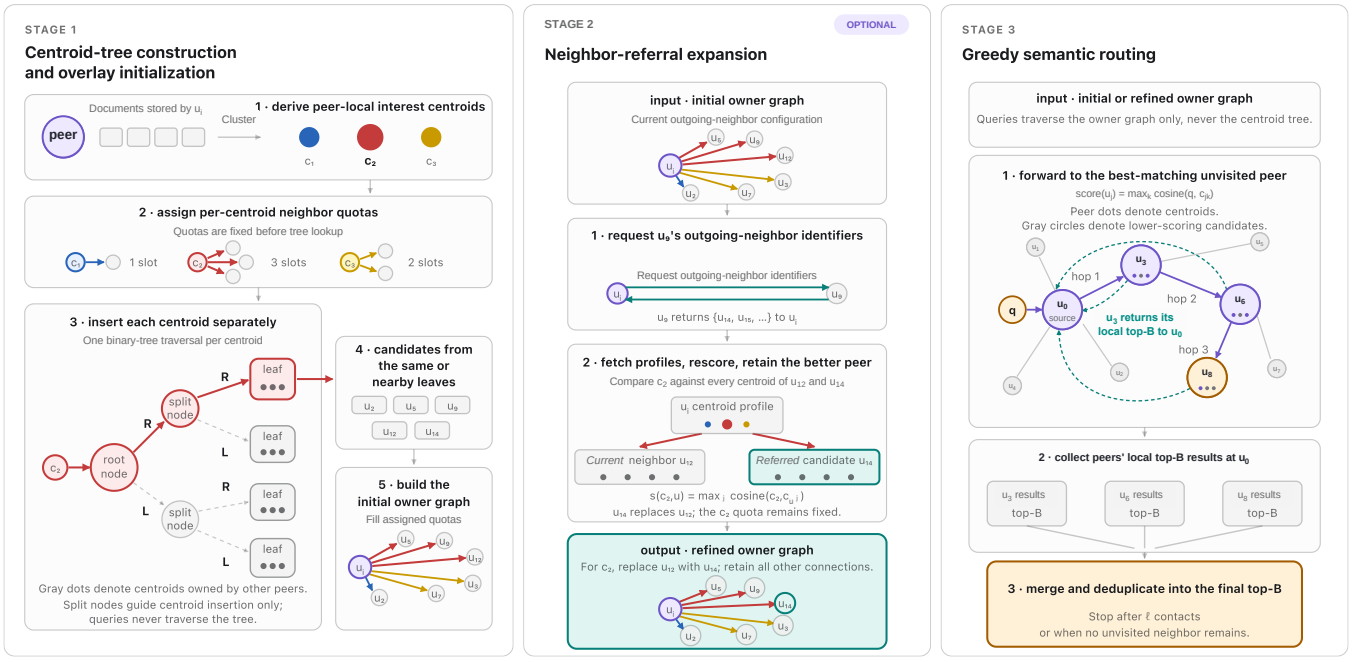


Figure 1: Overview of Semantica. Peer-interest centroids enter the tree, nearby-leaf candidates initialize the fixed-outdegree graph, referral rounds refine its links, and queries follow one greedy route while contacted peers return local rankings.

or recorded in a global registry. Documents and their embeddings remain with their owners, and there is no global map of identifiers.

The goal is to construct and search a directed semantic peer graph. The outgoing-neighbor set of u_i is $\mathcal{N}(u_i) \subseteq \mathcal{U} \setminus \{u_i\}$, with $|\mathcal{N}(u_i)| = d$ for a fixed $1 \le d < N$. A query consists of a normalized embedding $\mathbf{Q}$, a source peer $u_q \in \mathcal{U}$, a maximum number of peers to contact per query $0 \le \ell \le N-1$, and a result-list limit $B \ge 1$. We study remote retrieval: u_q initiates the search but its local collection is not searched. Let $\mathcal{D}_{-q} = \bigcup_{i \ne q} \mathcal{D}_i$, with copies sharing an identifier counted once. A document is therefore searchable by u_q when at least one peer other than u_q owns it. For a document x, let $s_{\mathbf{Q}}(x) = \mathbf{Q}^{\mathsf{T}}\mathbf{D}(x)$ denote its cosine similarity to the query. This score orders both the exhaustive reference ranking and the documents returned by contacted peers.

The reference ranking $E_B(\mathbf{Q})$ contains the first $\min\{B, |\mathcal{D}_{-q}|\}$ searchable documents ordered by decreasing similarity $s_{\mathbf{Q}}(x)$ and then by identifier. This corpus-wide ranking defines the retrieval objective but is unavailable to the search algorithm; it does not imply a global document index or owner directory.

Starting from u_q, the search discovers peers by traversing outgoing edges and contacts a set $\mathcal{S} \subseteq \mathcal{U} \setminus \{u_q\}$ with $|\mathcal{S}| \le \ell$. Documents become candidates only through responses from peers in $\mathcal{S}$. Each contacted peer u_i returns its local top B as identifier–score pairs, ordered by decreasing $s_{\mathbf{Q}}(x)$ and then by identifier. The source merges these pairs, removes duplicate identifiers, applies the same ordering to the returned scores, and retains at most B documents. We denote this merged ranking by $R_B(\mathbf{Q})$. The objective is for $R_B(\mathbf{Q})$ to approximate $E_B(\mathbf{Q})$ within the contact budget.

4 Design of Semantica

Figure 1 summarizes Semantica: the centroid tree initializes a fixed-outdegree semantic peer graph, the optional referral rounds refine the initial neighbor lists, and greedy routing lets the query source merge the local rankings returned by contacted peers into R_B.

4.1 Tree Construction

Each peer u_i embeds its documents $\mathcal{D}_i$ with the shared pre-trained model. Let $\mathbf{D}_{ij}$ be the normalized embedding of $d_{ij} \in \mathcal{D}_i$. Rather than using one mean, we divide the documents into k_i clusters whose normalized centroids $\mathbf{U}_{ic}$ represent separate interests:

$$\{\mathbf{U}_{i1}, \ldots, \mathbf{U}_{ik_i}\} = \text{normalize}\big(\text{kmeans}\left(\{\mathbf{D}_{ij} : d_{ij} \in \mathcal{D}_i\}, k_i\right)\big). \quad (1)$$

For peers u_i and u_p, we compare interest c of u_i with the closest interest of u_p; a query is compared with a peer in the same way:

$$s_{ic}(u_p) = \max_{1 \le h \le k_p} \mathbf{U}_{ic}^{\mathsf{T}}\mathbf{U}_{ph}, \qquad s_{\mathbf{Q}}(u_p) = \max_{1 \le h \le k_p} \mathbf{Q}^{\mathsf{T}}\mathbf{U}_{ph}. \quad (2)$$

Here i and p index peers, c and h index their interests, k_p is the number of interests retained by u_p, and $\mathbf{U}_{ic}$ and $\mathbf{U}_{ph}$ are the corresponding normalized centroids. The normalized vector $\mathbf{Q}$ represents a query. Thus $s_{ic}(u_p)$ is the greatest cosine similarity between source interest c and any interest of u_p, while $s_{\mathbf{Q}}(u_p)$ is the greatest similarity between the query and any interest of u_p.

Because the score takes a maximum over a peer's centroids, peers with more retained interests have more opportunities to obtain a high score and be forwarded the query.

Semantica accepts either a fixed centroid count or a selector that chooses it separately for each peer. The evaluation compares fixed counts $k_i \in \{1, \ldots, 8\}$ with chord-distance and cosine-silhouette

selectors. For the selectors, $k_i^{\max} = \min(8, |\mathcal{D}_i|, d)$; empty clusters are removed. The chord-distance selector, inspired by Satopaa et al. [23], normalizes the points $(k, \mathcal{I}_i(k))$, where $\mathcal{I}_i(k)$ is the K-means sum of squared distances. It chooses the point farthest from the line joining the first and last points, returning $k_i = 1$ when fewer than three values are available or the largest distance is at most 0.05. The silhouette selector uses cosine distance [22]: for document $\mathbf{x}$, $s(\mathbf{x}) = (b(\mathbf{x}) - a(\mathbf{x}))/\max\{a(\mathbf{x}), b(\mathbf{x})\}$, where a is its mean distance within its cluster and b the smallest mean distance to another cluster. It chooses the highest mean silhouette, using $k_i = 1$ if no score is defined and the smaller k on ties. In the evaluated implementation, K-means uses normalized inputs, a fixed seed per-peer, and `n_init=10`.

As an implementation choice, each peer keeps total outdegree d. It first assigns one outgoing slot to each retained centroid, then apportions the remaining $d - k_i$ slots in proportion to cluster size n_{ic} using the largest-remainder rule. Thus every centroid receives at least one outgoing slot and the allocations sum to d, keeping neighbor-list size independent of the number of centroids. This rule uses each cluster's share of documents as a proxy for future query share, which may fail if future query frequencies differ.

The tree stores centroid entries rather than documents. For interest c of peer u_i, let $e_{ic} = (u_i, c, \mathbf{U}_{ic})$; a distributed entry also carries its owner's network address.

The tree accepts entries sequentially and does not require a global insertion order.

A leaf n stores an ordered entry list E_n and is over capacity when $|E_n| > M$. An internal node n has children n_1, n_2 and stores their ordered pair of normalized split centers $(\boldsymbol{\mu}_{n1}, \boldsymbol{\mu}_{n2})$. An entry with vector $\mathbf{x}$ follows one branch according to

$$\rho_n(\mathbf{x}) = \begin{cases} n_1, & \|\mathbf{x} - \boldsymbol{\mu}_{n1}\|_2 < \|\mathbf{x} - \boldsymbol{\mu}_{n2}\|_2, \\ n_2, & \text{otherwise.} \end{cases} \tag{3}$$

Thus a descent tie goes to the second child. To split a leaf, let $\mathcal{X}_n$ be its set of distinct stored vectors and let $w_n(\mathbf{x})$ be the number of entries with vector $\mathbf{x}$:

$$\mathcal{X}_n = \{\mathbf{x}_e : e \in E_n\}, \qquad w_n(\mathbf{x}) = |\{e \in E_n : \mathbf{x}_e = \mathbf{x}\}|. \tag{4}$$

Exact duplicate vectors remain separate entries counting toward M, but form one weighted point during a split. Seeded weighted two-means fits two centers using

$$\Phi_n(\boldsymbol{\mu}_1, \boldsymbol{\mu}_2) = \sum_{\mathbf{x} \in \mathcal{X}_n} w_n(\mathbf{x}) \min_{r \in \{1,2\}} \|\mathbf{x} - \boldsymbol{\mu}_r\|_2^2. \tag{5}$$

The evaluated implementation uses ten initializations and the run seed, then normalizes both fitted centers. Each distinct vector, together with all of its entries, is redistributed by

$$\beta_n(\mathbf{x}) = \begin{cases} 1, & \|\mathbf{x} - \boldsymbol{\mu}_{n1}\|_2 \le \|\mathbf{x} - \boldsymbol{\mu}_{n2}\|_2, \\ 2, & \text{otherwise.} \end{cases} \tag{6}$$

A redistribution tie therefore goes to the first child, unlike a descent tie. An over-capacity leaf with at least two distinct vectors is eligible for splitting. If weighted two-means cannot produce two nonempty children, the leaf remains unchanged until the next entry to the leaf, when the split is retried.

Each tree node is stored by a selected peer, called its custodian. The bootstrap peer holds the empty root and publishes the protocol parameters and root descriptor. To insert an entry, its owner follows child descriptors under Equation 3 until reaching a leaf, then sends the entry to that leaf's custodian.

After a successful split, each child custodian is selected from the peers represented in that child. For child n_r, let

$$\mathcal{A}_r = \{u_i : \exists e_{ic} \in E_r\}, \qquad \kappa(n_r) = a_{r,j_r}, \tag{7}$$

where $a_{r,0}, \ldots, a_{r,m-1}$ are the members of $\mathcal{A}_r$ ordered by peer identifier, $m = |\mathcal{A}_r|$, and

$$j_r = H(\eta \mid s_r \mid \mathrm{id}(n_r) \mid a_{r,0} \mid \ldots \mid a_{r,m-1}) \bmod m. \tag{8}$$

Here H is SHA-256, η is the overlay epoch, and s_r is the parent's custody seed plus one. The selected peer recomputes the node identifier, candidate set, and selection before accepting custody. Each child custodian applies the same refinement rule to an over-capacity child before accepting it, and the original leaf becomes internal only after both children are accepted. Note that the K-means clustering depends on the population of the node which is splitting, so the final tree depends on insertion order.

Algorithm 1 Distributed insertion of one centroid entry

Input: Entry $e = (u_i, c, \mathbf{x})$, root n_{root}, capacity M, split seed s
Output: Entry e is present in its routed leaf

$n \leftarrow n_{\text{root}}$
while n is internal **do**
  $n \leftarrow \rho_n(\mathbf{x})$
end while
Append e to E_n at custodian $\kappa(n)$
if $|E_n| > M$ **then**
  $(\boldsymbol{\mu}_{n1}, \boldsymbol{\mu}_{n2}, E_1, E_2) \leftarrow$ WeightedSplit(E_n, s) using EQ. 4–6
  if $E_1, E_2 \neq \emptyset$ **then**
    for $r \in \{1, 2\}$ **do**
      Select and verify $\kappa(n_r)$ using EQ. 7–8
      $A_r \leftarrow$ CreateChild$_{\kappa(n_r)}(n, r, E_r)$
    end for
    await $A_1 = A_2 =$ accepted
    $n \leftarrow$ Internal$(\boldsymbol{\mu}_{n1}, \boldsymbol{\mu}_{n2}, n_1, n_2)$
  end if
end if

Algorithm 1 summarizes the construction. For each source centroid, a peer traverses to its initial leaf, retrieves profiles of the candidate peers represented in that leaf, and scores each candidate by its greatest centroid similarity to that source centroid. The peer orders the possible candidate–centroid assignments by decreasing similarity, breaking ties by increasing candidate-peer identifier and then increasing source-centroid index. It accepts an assignment only if the source centroid still has an unfilled outgoing slot and the candidate peer has not been assigned to another local centroid. For each centroid with unfilled slots, the peer adds candidates from every previously unseen leaf at the next distance from that centroid's initial leaf, then recomputes the joint assignment over all accumulated candidates. Only the completed initial outgoing-neighbor list is published; it contains d distinct remote peers.

4.2 Updating Outgoing Neighbors

Referral round r reads only the immediately preceding outgoing-neighbor lists: the initial lists when $r = 1$, and lists completed after referral round $r - 1$ otherwise. Peers advance independently. For each centroid with at least one assigned outgoing slot, the requester asks at most one previously unasked eligible referrer for the identifiers in that referrer's frozen preceding list. It prefers neighbors accepted during earlier referral rounds to neighbors from the initial graph and chooses pseudorandomly within the first nonempty group. The requester deduplicates the returned identifiers, excludes the peers in its own preceding list, then retrieves the remaining profiles from their owners. It processes centroids in increasing index order. For each centroid, it combines the new candidates with the outgoing neighbors assigned to that centroid and retains enough of the resulting ranking to fill the outgoing slots assigned to it, ordering candidates by decreasing similarity and then increasing remote-peer identifier; a remote peer retained for an earlier centroid is unavailable to later centroids. Outgoing neighbors accepted in referral round r become eligible referrers in referral round $r + 1$.

4.3 Greedy Semantic Routing

Starting from source u_q, which is added to the visited set but not searched, the query follows a single chain of peers. Each successive peer forwards the query to its highest-scoring unvisited outgoing neighbor using Equation 2, breaking ties by the smaller peer identifier. The search ends when no such neighbor remains or after ℓ remote peers have been contacted. Each contacted peer sends up to B pairs, of identifier and score, directly to the source and marks itself visited; if the search continues, the peer selects and forwards the query to the next peer. The source removes duplicate identifiers, orders the returned pairs by decreasing score and then increasing identifier, and retains at most B documents (our experiments use $B = 10$). When the limit is reached, the result of a search returns to the source without further contacts or comparisons.

5 Experimentation Methodology

`Semantica` requires three inputs for its evaluation: multi-interest peer histories, query–target relevance links, and embeddable query and document text. To our knowledge, no real-world data set exists that has these. This section details how we derive workloads from existing data sets and how we design our experiments using them.

5.1 Datasets

We construct workloads from three data sets: AOL-IA [15, 19], PubMedQA [12], and the Stack Overflow Duplicate Post Dataset [32]. $\mathcal{D}_i$ contains peer i's *training documents*, represented by its centroids; test queries search only these collections, using designated targets for evaluation.

Our first data set is *AOL-IA* and it contains user identifiers, query text and its timestamp, and the URL and rank of clicked documents. For AOL-IA, an event and its clicked result are removed if either lacks an embedding, leaving 9,089 users. For each user, randomly ordered unique user–document pairs are split into $\lfloor 2n/3 \rfloor$ training pairs and the test pairs. Documents are represented by their title, domain, and first 256 body words.

PubMedQA contains biomedical questions, abstract contexts, and yes/no/maybe answers. We treat each context as the target document, linked to the original question and five GPT-5.4 nano rephrasings as queries for it. For each of 50,000 simulated peers, 200 distinct target-document events are split into 133 training and 67 test events. Medical Subject Headings (MeSH) terms define K eligible topics. After m events, the process chooses a global topic with probability $K\alpha/(m + K\alpha)$, weighting topics by their queryable documents; otherwise it samples a previous topic in proportion to its frequency. A new document and an available query are then sampled from that topic. The main workload instance uses $\alpha = 5 \times 10^{-4}$, selected from topic-rank distributions without retrieval results, and has a median of 32 topics per peer. A separate concentrated-interest workload variant uses $\alpha = 3.5 \times 10^{-5}$ and a median of six topics.

The *Stack Overflow Duplicate Post Dataset* contains duplicate-question links. Documents consist of original titles and bodies; queries contain duplicate titles and bodies after removing duplicate notices, question links, anchor text, and bare URLs. Retaining documents linked to by at least two distinct query posts leaves 353,624 query posts, 65,836 targets, and 381,666 links. We simulate 50,000 peers from tags, sampling 200 distinct documents and one linked query per document before a 133/67 split. The query post is excluded from retrieval and the cosine reference. No Stack Overflow-specific tuning is used.

Embeddings. For all data sets, queries and documents use normalized `all-mpnet-base-v2` embeddings [21, 24, 26]. Queries are embedded separately, and no target embedding is substituted for a query embedding. For each seed in $\{42, 101, 202\}$, every method uses the same 100,000 set of events sampled uniformly without replacement; the seed fixes the chronological order and method randomness.

5.2 Baselines

We compare `Semantica` to five baselines: (i) P-stable locality-sensitive hashing (LSH), which assigns document embeddings to buckets of a prefix tree [5]; (ii) hierarchical navigable small-world graphs (HNSW), which search a layered proximity graph over document embeddings [16]; (iii) T-Man, which refines a peer graph by exchanging and reranking peer descriptions [11]; (iv) iCluster, which maintains a neighbor table for each peer interest and may branch a query across several links [20]; and (v) Graph-Diffusion, which summarizes each peer with one vector and diffuses (i.e., forwards) these summaries over an existing peer graph [9].

We use updated protocols for iCluster and T-Man (originally from 2008-2009), denoted as *adapted*, next to their *original* protocols. The adaptation makes them more compatible with modern document embeddings and, in case of T-man, it is the only way to evaluate them. Both adaptations use our peer representation and slot budget; a *peer descriptor* contains a peer identifier and its retained centroids.

For all baselines, we use the same per-seed query events, encoder, and union of peer training collections, with one item per distinct document identifier. Peer-overlay conditions also share document ownership. Semantica's greedy routing and result merging are applied unchanged to every graph. We now discuss the individual configurations of our baselines.

P-stable locality-sensitive hashing. P-stable LSH concatenates k_{LSH} hashes of width w in each of L tables, retrieves one matching bucket per table, excludes duplicate and source-only items, and cosine-ranks the remainder. The reported (k_{LSH}, L, w) settings are $(10, 64, 2)$, $(4, 30, 1)$, and $(8, 64, 2)$ for AOL-IA, PubMedQA, and Stack Overflow; configurations whose largest bucket exceeded 30% of the index were rejected. Querying more independent tables increases the chance that each document collides with the query and can make the returned candidate union approach the full collection, but incurs one logical bucket lookup per table, so a query performs L such lookups. For the illustrative Kademlia estimate, one DHT node per peer [17] and $\alpha = 3$ requests in each of exactly $\lceil \log_2 N \rceil$ stages per lookup give $M_{\text{Kad}} = 3L\lceil \log_2 N \rceil = O(L \log N)$ modeled request–reply exchanges, excluding final probes, returned-bucket payloads, retries, maintenance, and caching.

HNSW. An HNSW globally indexes the same distinct documents and their owners, using $M_H = 48$ and construction and requested search candidate-list sizes of 500. It is *centralized* because the index accesses the complete collection and owner lists rather than traversing owner peers. HNSW is not deployable under our no-global-index setting; its approximate result is not an upper bound, but the over-performance compared to the decentralized methods we consider to be the cost of decentralization.

Adapted T-Man. To operate, T-Man requires a similarity function (between peers): we provide this adaptation as follows. Each centroid excludes its four most recent partners and samples an exchange partner uniformly from the most-similar eligible half ($\psi = 0.5$). The peers exchange at most $\min\{d, 100\}$ descriptors and rerank the deduplicated candidates with Equation 2 while preserving per-centroid slot counts. We run twenty synchronous rounds from tree-derived and uniformly random graphs, then apply common routing.

Adapted/original iCluster. An interest table contains remote-peer descriptors assigned to one local centroid. Semantica's document-proportional allocation divides d slots among the tables; each labels approximately 90% of its most-similar entries *short-range* and the rest *long-range*. All thresholds share one uniformly random initial graph per seed.

Because iCluster can be tuned using a parameter θ, we additionally evaluate adapted iCluster using different parameterizations. Adapted iCluster allocates the shared d slots among centroid-specific interest tables, with approximately 90% short-range entries. From the same random graph per seed, we run twenty rounds for $\theta \in \{0.4, 0.5, 0.6, 0.7, 0.8\}$. We evaluate the resulting graphs under common routing and iCluster's own θ-controlled routing, capped at eight hops and 10,000 forwarded messages.

Graph-Diffusion. Graph-Diffusion uses Semantica's graph after twenty referral rounds but replaces its multi-centroid peer representation with one vector per peer. Each peer begins with the sum of its normalized training-document embeddings and iteratively mixes a λ-weighted copy of this vector with a $(1 - \lambda)$-weighted average of its outgoing neighbors' profiles. We compare the undiffused control ($\lambda = 1.0$) with Graph-Diffusion ($\lambda = 0.9$).

Semantica (this work). Each peer samples d distinct remote peers uniformly. Before referrals, comparing greedy routing with uniform selection among unvisited neighbors isolates the routing rule. Semantica's joint slot assignment partitions the same edges among centroids without changing the initial adjacency; applying the same referrals and routing to random and tree starts isolates initialization.

With evaluator-only access, this control compares every source centroid with every remote peer and applies Semantica's joint slot assignment to select 100 distinct outgoing peers. It uses no evaluation queries and is a graph-construction reference, not a retrieval upper bound or a guarantee of query-optimal routes.

5.3 Metrics

We have three main metrics that capture the information retrieval quality of any top-K results: *peer-overlap@K*, *target-hit@K*, and *nearest-found@K*. We discuss our main metrics in detail summarily. For each experiment, we also record the number of remote peers contacted, cosine comparisons used to score candidate peers, and unique peers and documents examined. Sending a query to one remote peer counts as one contact; replies, payload bytes, transport routes, and latency are excluded. Each cosine comparison used to score a candidate peer is counted separately.

Peer overlap. The peer overlap metric captures to what extent a peer's local connections match the top-K peers in the entire network. For each centroid of a peer's centroids ($c_i \in C$) we rank every other peer by the largest cosine similarity between c and any of that peer's centroids. The first K distinct remote peers form $E_K(c)$, and $A(c)$ is the set of outgoing peers assigned to c. We report the top-K peer overlap between these sets:

$$\text{peer-overlap@}K = \frac{1}{|C|} \sum_{c_i \in C} \frac{|A(c) \cap E_K(c)|}{\min\{K, |A(c)|\}}. \tag{9}$$

This metric lies on the range $[0, 1]$ and is higher as more of a peer's slots contain top-K peers. A higher score is better. We take $\min\{K, |A(c)|\}$ to prevent lower-ranked peers from gaining a higher score merely because a centroid has more slots. This metric is averaged over centroids without cluster-size weighting and describes the centroid space, not document retrieval or human relevance.

Target hit. The target hit metric, on the range $[0, 1]$, is simply the proportion of queries for which the dataset-designated target document appears among the first K returned documents after a given contact budget. A higher score is generally better. However, a lower score may be a result of a failure to find an available target or queries for which remote retrieval cannot return the target. Therefore, we additionally examine the proportion of queries whose route reaches an eligible non-source peer storing the target and the proportion of the queries whose target is stored by at least one peer other than the source.

Nearest found. Our final metric captures the ability of an algorithm to retrieve the nearest document that is available in the network. For a query embedding $\mathbf{Q}$, let $R_K(\mathbf{Q})$ and $E_K(\mathbf{Q})$ denote the first K documents in the returned ranking and the exhaustive reference ranking, respectively, as defined in Section 3:

$$\text{nearest-found@}K = \frac{|R_K(\mathbf{Q}) \cap E_K(\mathbf{Q})|}{K}. \tag{10}$$

Because the exhaustive reference ranking is induced by the shared encoder rather than independent human relevance judgments, nearest-found@K measures agreement with exhaustive encoder-based retrieval, not precision or human relevance.

Table 1: Nearest-found@10 and method-specific query-time operations.

Method	Nearest-found	Query-time operation
AOL-IA		
Semantica	57.94% ± 0.59	100 contacts
All-peer centroid comparison	58.74% ± 0.76	100 contacts
PPR vector ($\lambda = 0.9$)	44.87% ± 0.15	100 contacts
P-stable $(10, 64, 2)$	57.34% ± 1.18	64 / 2,688
HNSW	99.782% ± 0.005	centralized index query
PubMedQA		
Semantica	67.57% ± 0.05	100 contacts
All-peer centroid comparison	71.35% ± 0.04	100 contacts
PPR vector ($\lambda = 0.9$)	46.31% ± 0.15	100 contacts
P-stable $(4, 30, 1)$	68.76% ± 0.46	30 / 1,440
HNSW	99.99% ± 0.00	centralized index query
Adapted iCluster ($\theta = .4$)	85.97% ± 0.15	9,137 forwards
Adapted iCluster ($\theta = .5$)	55.18% ± 0.25	6,195 forwards
Adapted iCluster ($\theta = .7$)	0.923% ± 0.030	152 forwards
Adapted iCluster ($\theta = .8$)	0.018% ± 0.002	11 forwards
Stack Overflow		
Semantica	83.52% ± 0.15	100 contacts
All-peer centroid comparison	83.39% ± 0.14	100 contacts
PPR vector ($\lambda = 0.9$)	83.00% ± 0.07	100 contacts
P-stable $(8, 64, 2)$	84.81% ± 0.34	64 / 3,072
HNSW	100.00% ± 0.00	centralized index query

6 Evaluations

We perform three evaluations. First, we compare `Semantica` to our chosen baselines (see Section 5) w.r.t. its primary task of decentralized information retrieval. Second, we perform an ablation study of `Semantica`'s features. Lastly, we discuss the nuances of our initial results and provide follow-up experiments to highlight these.

Our configuration uses at most eight silhouette-selected centroids, outdegree and leaf capacity 100, and either zero or twenty referral rounds, with query budgets up to 100 contacts. This configuration was selected on PubMedQA and applied unchanged to AOL-IA and Stack Overflow.

6.1 Performance

Table 1 compares Semantica with the main retrieval baselines and the all-peer centroid comparison using Nearest-found@10 and a method-specific query-time operation. Nearest-found values are means over seeds 42, 101, and 202, followed by their sample standard deviations. The all-peer centroid comparison builds an outdegree-100 graph by comparing every source centroid with every remote peer and uses the same 100-contact query procedure as Semantica. Each P-stable row gives (k, L, w) and reports logical bucket lookups followed by modeled Kademlia request–reply exchanges; adapted-iCluster rows report forwarded query messages under the 10,000-message cap. On the two 50,000-peer workloads, the 100-contact budget visits at most 0.2% of the peer population.

The workloads span different levels of document replication. AOL-IA has 3.34 owners per searchable training document on average, PubMedQA has about 31, and Stack Overflow has about 101. Stack Overflow has the highest replication, yet even there an average document occurs at only about 0.20% of its 50,000 peers. Thus, document placement remains sparse even in the most replicated workload.

Relative to the primary Semantica graph after twenty referral rounds, constructing the graph by comparing every source centroid with every remote peer raises Nearest-found@10 by 0.80 percentage points on AOL-IA and 3.78 points on PubMedQA; the difference is positive in every seed. On Stack Overflow, this graph is lower by 0.13 points on average, with Semantica ahead by 0.11–0.15 points in each seed. The AOL-IA and Stack Overflow results show that Semantica nearly matches the Nearest-found obtained by comparing every source centroid with every remote peer when selecting outgoing neighbors. The Stack Overflow result is most likely the result of statistical noise rather than better wiring between peers.

Semantica contacts, adapted-iCluster forwarded query messages, and modeled Kademlia request–reply exchanges are compared as logical network interactions, not as packets, bytes, or latency; P-stable bucket lookups are reported separately alongside the corresponding Kademlia count. In every seed and on both retrieval measures, each setting with PPR diffusion ($\lambda < 1$) underperforms the single-vector setting without diffusion ($\lambda = 1$); the latter nevertheless remains below Semantica on all three workloads. HNSW reaches at least 99.78% nearest-found@10.

The proportion of queries whose route reaches a peer storing the target exceeds target hit@10 by 1.51 percentage points on PubMedQA, 26.13 on Stack Overflow, and 40.32 on AOL-IA. Reaching a peer that stores the target is almost sufficient for target hit@10 on PubMedQA, but not for historical clicks or duplicates. Stack Overflow credits only the designated duplicate, whereas an AOL-IA click need not rank among the encoder's nearest documents.

6.2 Ablations

The ablations below use PubMedQA unless specified otherwise.

6.2.1 Tree initialization and graph refinement. Figure 2 compares tree and random initialization: Figure 2 (a) shows Semantica before referrals and after twenty referral rounds, while Figure 2 (b) shows T-Man with ($\psi = 0.5$) after twenty T-Man exchange rounds. Tree initialization gives higher Nearest-found@10 at every shared outdegree under both procedures. With Semantica's referrals, at outdegree 100 tree initialization exceeds random initialization by 8.85 percentage points. With T-Man, the tree-initialized graph reaches 32.36% Nearest-found at outdegree 10, compared with 23.82% from the random graph, an advantage of 8.54 points; at outdegree 100, the advantage is 1.45 points. T-Man exceeds Semantica's referral procedure at outdegree 100 by 2.18 points from tree initialization and by 9.58 points from random initialization. The narrowing gap at larger outdegree suggests that a stronger refinement procedure can compensate for weaker initialization with more outgoing connections, while the tree-initialized graph remains ahead throughout the tested range.

In Table 2, we show that the tree initialization starts with higher top-10 peer overlap on all three workloads and remains ahead after twenty referral rounds, when its advantage over random initialization is 8.10 percentage points on AOL-IA, 23.58 on PubMedQA, and 16.11 on Stack Overflow. The persistent advantage may arise because, once outgoing-neighbor lists become semantically clustered, a poorly matched initial contact can suggest few candidates relevant to the requester's centroid.

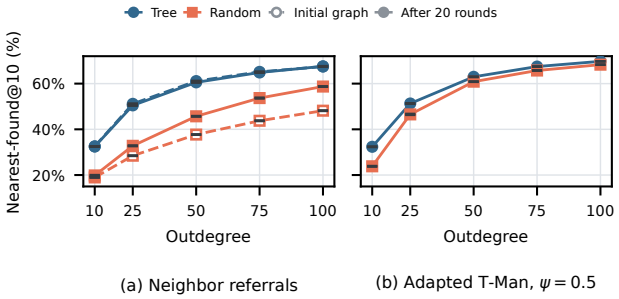


Figure 2: Nearest-found@10 by outdegree and initialization on PubMedQA.

Table 2: Top-10 peer overlap (%) at outdegree 100, before referrals and after twenty referral rounds (mean ± sample standard deviation over three seeds).

Workload	Initialization	Before	Round 20
AOL-IA	Tree	41.356 ± 0.344	89.110 ± 0.101
	Random	1.146 ± 0.009	81.015 ± 0.077
PubMedQA	Tree	22.668 ± 0.080	66.279 ± 0.086
	Random	0.189 ± 0.003	42.696 ± 0.081
Stack Overflow	Tree	25.628 ± 0.353	63.351 ± 0.192
	Random	0.196 ± 0.004	47.242 ± 0.153

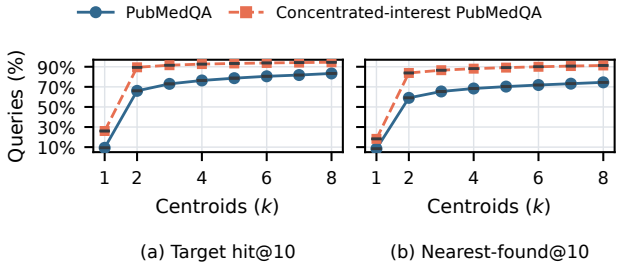


Figure 3: Retrieval across fixed centroid counts on two PubMedQA workload instances.

On the outdegree-100 random graph before referrals, greedy semantic routing yields 8.8 times the target hit@10 of uniform selection among unvisited neighbors at 100 contacts.

Referral rounds primarily improve short routes. Relative to the graph before referrals, twenty referral rounds add 4.62 target-hit@10 points on PubMedQA and 2.44 on Stack Overflow at a ten-contact budget, whereas the changes at 100 contacts are within 0.27 points across the three workloads. We therefore treat referral as an optional refinement; at the 100-contact budget, the tree-initialized graph already provides essentially the same retrieval quality.

6.2.2 Centroid count. Figure 3 compares retrieval across the fixed centroid counts. The $k = 1$ graphs consist of small disconnected components; from $k = 2$ onward, almost all peers form one strongly connected component. From two to eight, target hit still rises by 17.34 points on PubMedQA and 5.01 on concentrated-interest PubMedQA, so connectivity alone does not explain the gains. Fixed $k = 8$ is the upper endpoint of the tested range, not an identified optimum; it was fixed before the comparison results were examined. The reference silhouette method uses 4.45 and 3.60 centroids on average on PubMedQA and concentrated-interest PubMedQA, respectively. Silhouette is slightly higher on both retrieval measures for PubMedQA, whereas chord distance is higher by 0.78 target-hit points and 1.43 Nearest-found points on concentrated-interest PubMedQA. At $k = 8$, concentrated-interest PubMedQA is higher by 11.13 target-hit points and 16.80 Nearest-found points, but its independently generated ownership assignments and query events prevent a causal attribution to interest concentration.

6.2.3 Other topology parameters and diagnostics. After twenty referral rounds and at 100 contacts, outdegree 100 retains 94.2% of outdegree-200 target hit and 93.7% of its Nearest-found@10 while requiring 47.6% as many centroid comparisons during query routing. Across leaf capacities 10–200, target hit and Nearest-found@10 vary by 1.13 and 1.22 percentage points. Capacity 100 uses about fifteen times fewer leaves than capacity 10; capacity 200 halves the leaf count again but approximately doubles the number of centroid comparisons required to construct the initial graph. Compared with allowing every outgoing edge to be replaced, keeping some randomly selected edges unchanged improves target hit and Nearest-found@10 by at most 0.21 and 0.39 points while increasing referral requests by 18%. We retain outdegree 100 and leaf capacity 100 and allow referrals to replace every outgoing edge, though we do not consider these values optima.

With seed 42, the three PubMedQA insertion orders differ by 0.08 points in target hit@10 and 0.07 points in Nearest-found@10 after twenty referral rounds at 100 contacts. In each workload's seed-42 graph after twenty referral rounds, indegree has a Gini coefficient between 0.397 and 0.501; the highest-indegree 1% of peers receive 5.49–8.30% of edges. The tested insertion-order changes therefore have little effect on seed-42 retrieval, while the graphs after twenty referral rounds exhibit indegree concentration.

6.2.4 Adapted-iCluster graph ablation. Figure 5 (a) holds training-document assignments, centroids, query events, and greedy semantic routing fixed, so only graph organization differs. Semantica has higher Nearest-found@10 at every nonzero budget and every tested adapted-iCluster threshold. Against the highest-Nearest-found adapted-iCluster setting ($\theta = 0.8$), its Nearest-found@10 advantage is 7.32 points at ten contacts and 6.40 at 100 contacts. At 100 contacts, its target hit is 5.58 percentage points higher, and the proportion of its routes reaching a peer that stores the target is 5.84 points higher. Relative to the initial Semantica graph, twenty referral rounds add 2.86 Nearest-found points at ten contacts but 0.10 at 100 contacts. Twenty referral rounds raise mean top-10 peer overlap by 47.75 points on AOL-IA, 43.61 on PubMedQA, and 37.72 on Stack Overflow. On PubMedQA, adapted iCluster reaches 0.194% ± 0.003 at $\theta = 0.4$ and 2.300% ± 0.004 at $\theta = 0.8$. Semantica's overlap after twenty rounds is 28.8 times the latter value; before referrals it is 9.9 times that.

6.3 Discussion and Follow-up Experiments

Table 1 shows that centralized HNSW recovers at least 99.78% of the exact top ten on every workload. The reported setting maximizes Nearest-found@10 over the tested grid and uses $M_H = 48$,

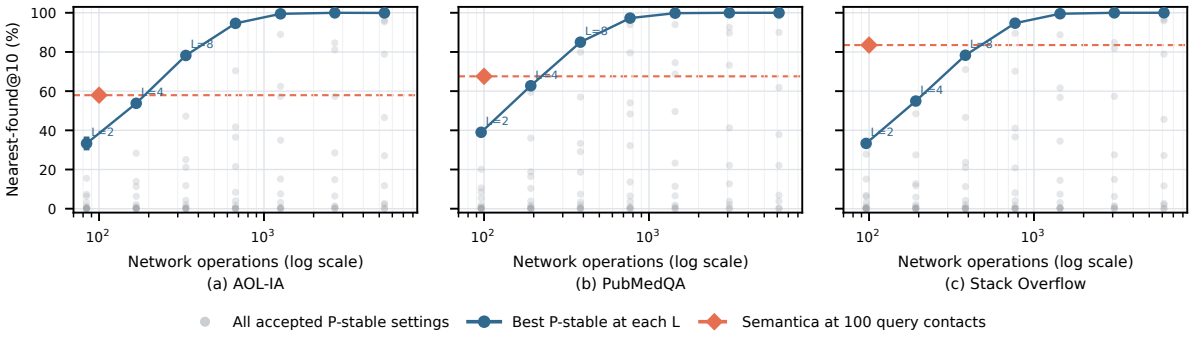


Figure 4: P-stable Nearest-found@10 under the Kademlia lookup model; dotted lines show Semantica after 20 referral rounds.

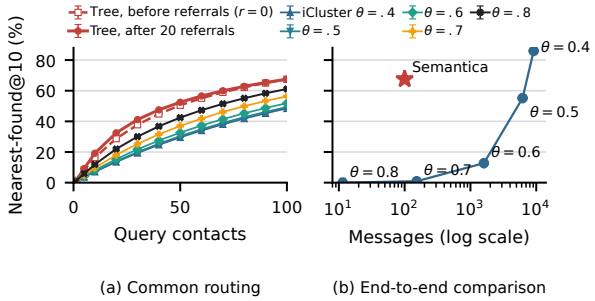


Figure 5: Adapted iCluster on PubMedQA when all graphs use Semantica's routing (a) and when each method uses its own routing procedure (b).

`ef_construction`=500, and `ef_search`=500. In separate target-hit measurements, HNSW is 0.18 percentage points below Semantica on AOL-IA, 21.07 points above it on PubMedQA, and 2.78 points above it on Stack Overflow. Target hit can differ from Nearest-found@10 because the target need not be among the cosine neighbors.

The P-stable rows in Table 1 were selected post hoc from the completed grid among settings that retrieve the matching bucket from every table; they are the settings closest to Semantica rather than independently selected choices. For each L, Figure 4 reports the maximum mean Nearest-found among configurations completed for all three seeds and omits a configuration if any seed rejected it for an oversized bucket. Settings with $L = 2$ remain below Semantica on all workloads. At $L = 4$, they come within about five percentage points on AOL-IA and PubMedQA. Settings with $L = 8$ exceed Semantica on those workloads but not Stack Overflow while evaluating cosine similarity against roughly two-fifths of each corpus. At the selected comparable-quality operating points, Semantica obtains similar nearest-found@10 with 100 peer contacts, whereas our illustrative Kademlia realization of P-stable LSH incurs 1,440–3,072 routing request–reply exchanges—14.4–30.7 times as many logical network interactions—even before terminal bucket transfer and distributed candidate access are counted.

Adapted iCluster. Figure 5 (a) applies Semantica's greedy routing and document ranking to every graph, isolating graph organization. Figure 5 (b) compares the methods when each uses its own query procedure. Both cases include every tested adapted-iCluster threshold. At $\theta = 0.5$, Semantica obtains 67.57% ± 0.05 Nearest-found@10 with 100 contacts, whereas adapted iCluster obtains 55.18% ± 0.25 with 6,195 ± 29 forwarded messages; 58.52% of queries reach the 10,000-message cap. Semantica is 12.39 percentage points higher. Each forward carries the query embedding, so the forwarding count also reflects the repeated query-payload transmission. Concurrency may reduce latency, but not aggregate traffic. Among the adapted-iCluster thresholds tested, $\theta = 0.5$ gives the Nearest-found closest to Semantica. The comparison uses the same per-seed queries, training documents, fitted centroids, source exclusion, and encoder-defined top-10 reference set; graph construction and query routing differ. At $\theta = 0.4$, adapted iCluster reaches 85.97% ± 0.15 with 9,137 ± 12 forwarded messages, and 90.36% of queries reach the cap. For capped queries, evaluation retains all documents returned before forwarding stops. This capped setting does not characterize the full range of Nearest-found and message counts achievable by adapted iCluster.

7 Complexity Analysis

Let $C = \sum_i k_i$ be the number of centroid entries, K the configured upper bound on k_i, d the peer outdegree, R the number of referral rounds, Q the number of queries, ℓ the contact budget, and b the embedding dimension. Thus $k_i \le K$ for every peer. For a peer with m_i training documents, fitting k_i centroids for I K-means iterations costs $O(m_i k_i I b)$. Fitting every candidate count through K costs $O(m_i I b K^2)$, while evaluating cosine silhouette through that limit can add $O(K m_i^2 b)$ in the worst case.

Inserting all C centroids costs $\Theta(Cb \log C)$ for a balanced tree and $\Theta(C^2 b)$ for a pathologically skewed tree. With fixed leaf capacity, recursive two-means splitting adds $O(CIb \log C)$ for a balanced tree and $O(C^2 I b)$ in the maximally skewed case. If A_t is the number of candidate–centroid pairs accumulated after tree-distance level t and A their total, neighbor assignment costs $O(KAb + \sum_t A_t \log A_t)$.

During each referral round, each peer sends at most one neighbor-list request per active centroid, giving $O(RC)$ such requests and $O(RC(Kdb + d \log d))$ local work; fetching profiles for previously unknown suggestions adds protocol requests proportional to the distinct candidates returned. Greedy semantic routing contacts at most ℓ remote peers per query. If peer v_h, contacted at step h, stores

m_{v_h} documents, one query costs $O\left(\sum_{h=1}^{\ell} b(Kd + m_{v_h})\right)$. Each contacted peer also sends one result response directly to the source, whether or not it forwards the query.

8 Conclusion

Our proposed system, `Semantica`, is built around a simple—yet effective—novel idea: insert peers' interest centroids into a centroid tree, then use peers associated with nearby tree entries to initialize a semantic overlay. Peers can subsequently refine their outgoing connections through referrals to other yet-unconnected peers, shortening retrieval paths. Our ablations have shown that our centroid tree initialization achieves higher retrieval metrics when compared to random initialization under both tested methods for updating neighbors, as well as other initial graphs constructions, in a controlled comparison that held the routing procedure fixed. Across three workloads, our referral produces large retrieval gains with a small amount of network communication. These results suggest that centroid trees are a promising foundation for building semantic overlays in decentralized search systems.

9 Ethical Considerations and Limitations

Decentralized search is in part an attempt to address concerns with centralized search, including the concentration of influence over what can be found [10] and the privacy risks that arise when a single provider logs queries and builds user profiles [1, 28]. But decentralized search on its own does not fully resolve all concerns. We therefore note the ethical considerations a decentralized deployment should also keep in mind.

For example, without implementing additional measures, deployment can expose queries, interest centroids, and neighbor identifiers that could support profiling. Access control, short-lived peer identifiers, private similarity protocols, or perturbed centroids could reduce exposure (at possible retrieval cost). Private profile comparison has been studied in the literature for semantic overlays [35]. Malicious peers could also advertise false centroids, poison candidates, suppress results, or overload the unreplicated tree custodians, alongside the Sybil, Eclipse, routing, and storage attacks faced by open DHTs [30]. Because the root custodian must remain reachable and node custody is not reassigned after failure, deployment would require rate limits, replicated and reassignable custody, validation, and exclusion mechanisms, which we did not evaluate in this paper.

Further, we acknowledge that AOL-IA is a public but sensitive historical log. We use only the standard released collection, do not re-host the raw queries, and report aggregate results rather than individual query histories.

Finally, we also note limitations of our analysis that may be relevant to ethical considerations for deployment and for future analysis. For example, large-scale evaluation uses a static in-memory simulation; a 100-peer TCP implementation checks functional equivalence, but not deployment performance or behavior under churn; that would require experiments with real users which are beyond the scope of this paper. PubMedQA and Stack Overflow use simulated peers, and AOL-IA does not use a chronological split. Therefore, designated-target and encoder-neighbor measures do not necessarily establish human relevance. Finally, since the adaptive centroid score may favor peers with more centroids, while document-proportional interest quotas assume that larger clusters warrant more outgoing slots, we therefore do not attribute retrieval gains to unequal allocation.

AI coding assistants were used while implementing a few of the experiments in the paper. Any such code was carefully reviewed.

Code availability. The source code used in experiments is available in an anonymous repository at https://anonymous.4open.science/r/Semantica-15E3/README.md.

References

[1] Michael Barbaro, Tom Zeller, and Saul Hansell. 2006. A face is exposed for AOL searcher no. 4417749. *New York Times* 9, 2008 (2006), 8. Retrieved August 25, 2026 from https://www.nytimes.com/2006/08/09/technology/09aol.html

[2] Sergey Brin and Lawrence Page. 1998. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems* 30, 1-7 (1998), 107–117. doi:10.1016/S0169-7552(98)00110-X

[3] Andrea Cimmino, Michael McCool, Farshid Tavakolizadeh, and Kunihiko Toumura. 2023. *Web of Things (WoT) Discovery*. W3C Recommendation. World Wide Web Consortium. https://www.w3.org/TR/2023/REC-wot-discovery-20231205/

[4] Arturo Crespo and Hector Garcia-Molina. 2005. Semantic Overlay Networks for P2P Systems. In *Agents and Peer-to-Peer Computing (Lecture Notes in Computer Science, Vol. 3601)*, Gianluca Moro, Sonia Bergamaschi, and Karl Aberer (Eds.). Springer, 1–13. doi:10.1007/11574781_1

[5] Mayur Datar, Nicole Immorlica, Piotr Indyk, and Vahab S. Mirrokni. 2004. Locality-Sensitive Hashing Scheme Based on p-Stable Distributions. In *Proceedings of the Twentieth Annual Symposium on Computational Geometry*. 253–262. doi:10.1145/997817.997857

[6] Shiyuan Deng, Xiao Yan, Kelvin K. W. Ng, Chenyu Jiang, and James Cheng. 2019. Pyramid: A General Framework for Distributed Similarity Search on Large-Scale Datasets. In *2019 IEEE International Conference on Big Data (Big Data)*. IEEE, 1066–1071. doi:10.1109/BigData47090.2019.9006219

[7] Wei Dong, Moses Charikar, and Kai Li. 2011. Efficient K-Nearest Neighbor Graph Construction for Generic Similarity Measures. In *Proceedings of the 20th International Conference on World Wide Web*. Association for Computing Machinery, 577–586. doi:10.1145/1963405.1963487

[8] Christos Doulkeridis, Kjetil Nørvåg, and Michalis Vazirgiannis. 2008. Peer-to-Peer Similarity Search over Widely Distributed Document Collections. In *Proceedings of the 2008 ACM Workshop on Large-Scale Distributed Systems for Information Retrieval*. Association for Computing Machinery, Napa Valley, CA, USA, 35–42. doi:10.1145/1458469.1458477

[9] Nikolaos Giatsoglou, Emmanouil Krasanakis, Symeon Papadopoulos, and Ioannis Kompatsiaris. 2022. A Graph Diffusion Scheme for Decentralized Content Search Based on Personalized PageRank. In *2022 IEEE 42nd International Conference on Distributed Computing Systems Workshops*. 53–59. doi:10.1109/ICDCSW56584.2022.00019

[10] Lucas D Introna and Helen Nissenbaum. 2000. Shaping the Web: Why the politics of search engines matters. *The information society* 16, 3 (2000), 169–185. doi:10.1080/01972240050133634

[11] Márk Jelasity, Alberto Montresor, and Özalp Babaoğlu. 2009. T-Man: Gossip-Based Fast Overlay Topology Construction. *Computer Networks* 53, 13 (2009), 2321–2339. doi:10.1016/j.comnet.2009.03.013

[12] Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. 2019. PubMedQA: A Dataset for Biomedical Research Question Answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, Hong Kong, China, 2567–2577. doi:10.18653/v1/D19-1259

[13] Naama Kraus, David Carmel, Idit Keidar, and Meni Orenbach. 2016. NearBucket-LSH: Efficient Similarity Search in P2P Networks. In *Similarity Search and Applications (Lecture Notes in Computer Science, Vol. 9939)*, Laurent Amsaleg, Michael E. Houle, and Erich Schubert (Eds.). Springer, 236–249. doi:10.1007/978-3-319-46759-7_18

[14] Mei Li, Wang-Chien Lee, Anand Sivasubramaniam, and Jing Zhao. 2008. SSW: A Small-World-Based Overlay for Peer-to-Peer Search. *IEEE Transactions on Parallel and Distributed Systems* 19, 6 (2008), 735–749. doi:10.1109/TPDS.2007.70757

[15] Sean MacAvaney, Craig Macdonald, and Iadh Ounis. 2022. Reproducing Personalised Session Search over the AOL Query Log. In *Advances in Information Retrieval: 44th European Conference on IR Research, ECIR 2022, Stavanger, Norway, April 10–14, 2022, Proceedings, Part I (Lecture Notes in Computer Science, Vol. 13185)*, Matthias Hagen, Suzan Verberne, Craig Macdonald, Christin Seifert, Krisztian Balog, Kjetil Nørvåg, and Vinay Setty (Eds.). Springer, 627–640. doi:10.1007/978-3-030-99736-6_42

[16] Yu. A. Malkov and D. A. Yashunin. 2020. Efficient and Robust Approximate Nearest Neighbor Search Using Hierarchical Navigable Small World Graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42, 4 (2020), 824–836. doi:10.1109/TPAMI.2018.2889473

[17] Petar Maymounkov and David Mazières. 2002. Kademlia: A Peer-to-Peer Information System Based on the XOR Metric. In *Peer-to-Peer Systems (Lecture Notes in Computer Science, Vol. 2429)*. Springer, 53–65. doi:10.1007/3-540-45748-8_5

[18] Petru Neague, Marcel Gregoriadis, and Johan Pouwelse. 2024. De-DSI: Decentralised Differentiable Search Index. In *Proceedings of the 4th Workshop on Machine Learning and Systems*. Association for Computing Machinery, 134–143. doi:10.1145/3642970.3655837

[19] Greg Pass, Abdur Chowdhury, and Cayley Torgeson. 2006. A Picture of Search. In *Proceedings of the 1st International Conference on Scalable Information Systems (ACM International Conference Proceeding Series, Vol. 152)*. Association for Computing Machinery, 1–es. doi:10.1145/1146847.1146848

[20] Paraskevi Raftopoulou and Euripides G. M. Petrakis. 2008. iCluster: A Self-Organizing Overlay Network for P2P Information Retrieval. In *Advances in Information Retrieval: 30th European Conference on IR Research (Lecture Notes in Computer Science, Vol. 4956)*. Springer, 65–76. doi:10.1007/978-3-540-78646-7_9

[21] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, Hong Kong, China, 3982–3992. doi:10.18653/v1/D19-1410

[22] Peter J. Rousseeuw. 1987. Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis. *J. Comput. Appl. Math.* 20 (1987), 53–65. doi:10.1016/0377-0427(87)90125-7

[23] Ville Satopaa, Jeannie Albrecht, David Irwin, and Barath Raghavan. 2011. Finding a "Kneedle" in a Haystack: Detecting Knee Points in System Behavior. In *2011 31st International Conference on Distributed Computing Systems Workshops*. IEEE, 166–171. doi:10.1109/ICDCSW.2011.20

[24] Sentence Transformers. 2022. all-mpnet-base-v2 Model Card. Hugging Face. https://huggingface.co/sentence-transformers/all-mpnet-base-v2/tree/e8c3b32edf5434bc2275fc9bab85f82640a19130 Repository revision e8c3b32; accessed 2026-08-17.

[25] Milad Shokouhi and Luo Si. 2011. Federated search. *Foundations and trends® in information retrieval* 5, 1 (2011), 1–102. doi:10.1561/1500000010

[26] Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. MPNet: Masked and Permuted Pre-training for Language Understanding. In *Advances in Neural Information Processing Systems*, Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (Eds.), Vol. 33. Curran Associates, Inc., 16857–16867. https://proceedings.neurips.cc/paper/2020/hash/c3a690be93aa602ee2dc0ccab5b7b67e-Abstract.html

[27] Chunqiang Tang, Zhichen Xu, and Sandhya Dwarkadas. 2003. Peer-to-Peer Information Retrieval Using Self-Organizing Semantic Overlay Networks. In *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*. 175–186. doi:10.1145/863955.863976

[28] Omer Tene. 2008. What google knows: Privacy and internet search engines. *Utah L. Rev.* (2008), 1433. doi:10.63140/eatu6hcsq2

[29] Duc A. Tran. 2005. A Hierarchical Semantic Overlay Approach to P2P Similarity Search. In *2005 USENIX Annual Technical Conference (USENIX ATC 05)*. USENIX Association, Anaheim, CA. https://www.usenix.org/conference/2005-usenix-annual-technical-conference/hierarchical-semantic-overlay-approach-p2p

[30] Guido Urdaneta, Guillaume Pierre, and Maarten van Steen. 2011. A Survey of DHT Security Techniques. *Comput. Surveys* 43, 2, Article 8 (2011), 49 pages. doi:10.1145/1883612.1883615

[31] Shuai Wang, Ekaterina Khramtsova, Shengyao Zhuang, and Guido Zuccon. 2024. FeB4RAG: Evaluating Federated Search in the Context of Retrieval Augmented Generation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. Association for Computing Machinery, 763–773. doi:10.1145/3626772.3657853

[32] Xingfang Wu, Heng Li, Nobukazu Yoshioka, Hironori Washizaki, and Foutse Khomh. 2024. Refining GPT-3 Embeddings with a Siamese Structure for Technical Post Duplicate Detection. In *2024 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*. 114–125. doi:10.1109/SANER60148.2024.00019 Associated data set: https://doi.org/10.5281/zenodo.10067702.

[33] Chenhao Xu, Longxiang Gao, Yuan Miao, and Xi Zheng. 2025. Distributed Retrieval-Augmented Generation. doi:10.48550/arXiv.2505.00443 arXiv:2505.00443 [cs.DC]

[34] Tarek Zaarour and Edward Curry. 2022. SemanticPeer: A Distributional Semantic Peer-to-Peer Lookup Protocol for Large Content Spaces at Internet-Scale. *Future Generation Computer Systems* 132 (2022), 239–253. doi:10.1016/j.future.2022.02.016

[35] Niels Zeilemaker, Zekeriya Erkin, Paolo Palmieri, and Johan Pouwelse. 2013. Building a Privacy-Preserving Semantic Overlay for Peer-to-Peer Networks. In *2013 IEEE International Workshop on Information Forensics and Security*. IEEE, 79–84. doi:10.1109/WIFS.2013.6707798

[36] Hela Zorgati, Raoudha Ben Djemaa, and Ikram Amous. 2023. Efficient IoT Resource Discovery Approach Based on P2P Networks and Fog Computing. *Internet of Things* 24 (2023), 100954. doi:10.1016/j.iot.2023.100954