Peer-to-Peer Information Retrieval Using Self-Organizing Semantic Overlay Networks

Chunqiang Tang, Zhichen Xu, and Sandhya Dwarkadas
SIGCOMM 2003
Presented by Keith Tayloe

Peer-to-Peer Information Retrieval
- Distributed Hash Table (DHT)
  - CAN, Chord, Pastry, Tapestry, etc.
  - Scalable, fault tolerant, self-organizing
  - Only support exact key match
    - \( K_d = \text{hash} \left( \text{"books on computer networks"} \right) \)
    - \( K_s = \text{hash} \left( \text{"computer network"} \right) \)
- Extend DHTs with content-based search
- Full-text search, music/image retrieval
- Build large-scale search engines using P2P technology

Focus and Approach in pSearch
- Efficiency
  - Search a small number of nodes
  - Transmit a small amount of data
- Efficacy
  - Search results comparable to centralized information retrieval (IR) systems
  - Extend classical IR algorithms to work in DHTs, both efficiently and effectively

Outline
- Key idea in pSearch
- Background
  - Information Retrieval (IR)
  - Content-Addressable Network (CAN)
- Our P2P IR algorithm
- Experimental results
- Open issues and ongoing work
- Conclusions
**Outline**

- Key idea in pSearch
- Background
  - Information Retrieval (IR)
  - Content-Addressable Network (CAN)
- Our P2P IR algorithm
- Experimental results
- Open issues and ongoing work
- Conclusions
Background

- Statistical IR algorithms
  - Vector Space Model (VSM) [Salton et al.]
  - Latent Semantic Indexing (LSI) [Deerwester et al.]
- Distributed Hash Table (DHT)
  - Content-Addressable Network (CAN) [Ratnasamy et al.]

Background: Vector Space Model

**Example query** = *baking*

$q^{(1)} = (1 \ 0 \ 0 \ 0 \ 0 \ 0)^T$

- Search for relevant documents is carried out by computing the cosines of the angles $\theta_j$ between the query vector $q^{(1)}$ and the document vectors $a_j$
- Results: only nonzero cosines are $\cos \theta_i = 0.5774$ and $\cos \theta_4 = 0.4082$
**Background: Latent Semantic Indexing**

- **documents**
- **semantic vectors**
- **SVD:** singular value decomposition
  - Reduce dimensionality
  - Suppress noise
  - Discover word semantics
    - Car <-> Automobile

**Background: Content-Addressable Network**

- Partition Cartesian space into zones
- Each zone is assigned to a computer
- Neighboring zones are routing neighbors
- An object key is a point in the space
- Object lookup is done through routing

---

**Outline**

- Key idea in pSearch
- Background
  - Information Retrieval (IR)
  - Content-Addressable Network (CAN)
- Our P2P IR algorithm
- Experimental results
- Open issues and ongoing work
- Conclusions

**pLSI Basic Idea**

- Use a CAN to organize nodes into an overlay
- Use semantic vectors generated by LSI as object key to store doc indices in the CAN
  - Index locality: indices stored close in the overlay are also close in semantics
- Two types of operations
  - Publish document indices
  - Process queries
pLSI Illustration

Major Challenges
- Dimensionality mismatch between CAN and LSI
  - Large search space
- The curse of dimensionality
  - Inefficient searching
- Uneven distribution of document indices
  - Inefficient routing and unbalanced load

pLSI Enhancements
- Further reduce nodes visited during a search
  - Multi-plane (Rolling-index)
  - Content-directed search
- Balance index distribution
  - Content-aware node bootstrapping

Multi-plane (rolling index)
- 4-d semantic vectors
Multi-plane (rolling index)

- 4-d semantic vectors
- 2-d CAN

21

Multi-plane (rolling index)

- 4-d semantic vectors
- 2-d CAN

22

Multi-plane (rolling index)

- 4-d semantic vectors
- 2-d CAN

23

Multi-plane (rolling index)

- 4-d semantic vectors
- 2-d CAN

24
Multi-plane (rolling index)
- 4-d semantic vectors
- 2-d CAN

Content-directed Search
- Search the node whose zone contains the query semantic vector. (query center node)

Content-directed Search
- Search direct (1-hop) neighbors of query center

Content-directed Search
- How about 2-hop neighbors of query center?
Content-directed Search

- Search direct (1-hop) neighbors; Selectively search some 2-hop neighbors
  - Focusing on "promising" regions suggested by samples

Content-Aware Node Bootstrapping

- pSearch randomly picks the semantic vector of an existing document for node bootstrapping

Outline

- Key idea in pSearch
- Background
  - Information Retrieval (IR)
  - Content-Addressable Network (CAN)
- Our P2P IR algorithm
- Experimental results
- Open issues and ongoing work
- Conclusions

Experiment Setup

- pSearch Prototype
  - Cornell’s SMART system implements VSM
  - We extended it with implementations of LSI, CAN, and our pLSI algorithms
- Corpus: Text Retrieval Conference (TREC)
  - 528,543 documents from various sources
  - total size about 2GB
  - 100 queries, topic 351-450
Evaluation Metrics

- Efficiency: nodes visited and data transmitted during a search
- Efficacy: compare search results
  - pLSI vs. LSI
  - pLSI vs. best known IR algorithms

pLSI vs. LSI

\[
\text{Accuracy} = \frac{|A \cap B|}{|A|} \times 100\%
\]

- Retrieve top 15 documents
- A: documents retrieved by LSI
- B: documents retrieved by pLSI

Performance w.r.t. System Size

- \text{Accuracy} = 90\%
- Search < 0.2\% nodes
- Transmit 72KB data

Performance & System Size

- Relaxing Quit Bound improves accuracy slowly with No. of nodes visited.
- Suggest results can be returned w/o waiting for query to reach final bound
Performance & Replication

- Accuracy of Content can approach 90% @ .2% of nodes
- With replication and query heuristics can achieve 91.7% @ 19 nodes or 98% at 45 nodes.

Open Issues & Ongoing Work

- Larger corpora, other docs or queries
- Efficient variants of LSI/SVD: 1 hour->1min
- Evolution of global statistics
- Incorporate other IR techniques
  - Relevance feedback, Google's PageRank, Music and image retrieval
- Compare with other alternatives
  - pVSM [Tang et al., HotNets-I]

Conclusion

- We map semantic space generated by modern IR algorithms atop overlay networks to enable efficient P2P search
  - pLSI is good at clustering documents
  - Index locality: indices stored close in the overlay network are also close in semantics
- We introduced techniques to
  - Further reduce visited nodes: content-directed search & rolling index
  - Balance index distribution: content-aware node bootstrapping