A general framework for content search in P2P networks is proposed
Based on the framework, we implement a semantic-based document search system
The Framework

- Underlying P2P architecture – SuperNode network;
- Hierarchical summary structure (three levels)
  - *Unit Level (the lowest level)* – an information unit, such as a document or an image, is summarized;
  - *Peer Level (the second level)* – all information in a peer is summarized;
  - *Super Level (the third level)* – all information contained by a peer group is summarized;
The Framework

- Each super-peer maintains two pieces of summaries:
  - super level summaries of its group & its neighboring groups;
  - peer level summaries of its group;
- Indexes are built on summaries. Accordingly, three kinds of indexes are maintained:
  - Local index --- for unit level summaries;
  - Group index --- for peer level summaries;
  - Global index --- for super level summaries;
The summarization method is domain specific. All three levels may use same or different summarization methods. So are index methods.

By the framework, information searching can become more guided: a peer group is first decided; then a peer; and finally, an information unit.
Suppose there are a large number of peers in the network, and each peer contains a large number of documents, what we want to achieve is to find the most relevant documents as quickly as possible, given a query (keywords, or a sentence).
The system is built on the above framework;

Summary Building

- The summarization is also done in three levels;
- For each level, there are two steps: VSM, LSI;
- VSM (Vector Space Model)
  - Each document is represented by a vector of weighted term frequencies.
  - Three factors are involved in term weighting: TF, IDF, and the normalization factor;
**LSI (Latent Semantic Indexing)**

- To discover the underlying semantic correlation among documents, overcoming synonymy, polysemy, and noise problems in information retrieval.
- A technique (SVD) is used to reduce the dimensional space.
- By this step, a very high-dimensional space (of tens of thousands) is reduced to a much smaller one (of less than two hundreds) to facilitate indexing & searching.
- Indexing summaries
  - A modified VA-file
    - VA-file outperform sequential scan in high-dimensional space;
    - VA-file is extremely computationally efficient for insertion;
    - VA-file is modified to search the nearest neighbors by similarity function:

\[
sim(Q, P) = \sum_{i=0}^{D-1} Q[i] \times P[i]
\]
Hierarchical Summarization/ Indexing
An Example

A small P2P network with 4 peer groups, each has 2 peers. Suppose there are only one document in each peer.

<table>
<thead>
<tr>
<th>Peer</th>
<th>Document</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>Monitoring XML Data on the Web</td>
</tr>
<tr>
<td>P2</td>
<td>Approximate XML Joins</td>
</tr>
<tr>
<td>P3</td>
<td>Outlier Detection for High Dimensional Data</td>
</tr>
<tr>
<td>P4</td>
<td>High Dimensional Indexing Using Sampling</td>
</tr>
<tr>
<td>P5</td>
<td>Document Clustering with Committees</td>
</tr>
<tr>
<td>P6</td>
<td>Document Clustering with Cluster Refinement</td>
</tr>
<tr>
<td>P7</td>
<td>The Language Model for Information Retrieval</td>
</tr>
<tr>
<td>P8</td>
<td>Document Summarization in Information Retrieval</td>
</tr>
</tbody>
</table>
Monitoring XML Data on the Web

Peer dictionary

Super Peer 1

Peer 1

Peer 2

The Approximate XML Joins

Peer document

Peer dictionary

Peer document

Much lower dimensional points

Peer1: (1.83, 1.13)
Peer2: (0.8, -1.31)

VSM:

<table>
<thead>
<tr>
<th></th>
<th>P1</th>
<th>P2</th>
</tr>
</thead>
<tbody>
<tr>
<td>w1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>w2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>w3</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>w4</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>w5</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>w6</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

SVD

VA file
### Global VSM

<table>
<thead>
<tr>
<th></th>
<th>grp1</th>
<th>grp2</th>
<th>grp3</th>
<th>grp4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Approximate</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Cluster</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>Committee</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Data</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Detection</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Dimension</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Document</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>High</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Index</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Information</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Join</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Language</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Model</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Monitor</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Outlier</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Refinement</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Retrieval</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Sampling</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Summarize</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Title</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Using</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Web</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>XML</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

#### High dimensional points

- **Group1**: $(0.71, 3.67, 0, 0)$
- **Group2**: $(0, 0, 2.68, 1.33)$
- **Group3**: $(0, 0, 1.34, -2.65)$
- **Group4**: $(-3.70, 0.7, 0, 0)$

---

**Global dictionary**

- Approximate
- Cluster
- Committee
- Data
- Detection
- Dimension
- Document
- High
- Index
- Information
- Join
- Language
- Model
- Monitor
- Outlier
- Refinement
- Retrieval
- Sampling
- Summarize
- Title
- Using
- Web
- XML

**VA file**
Query Processing

• When a peer issues a query, it is passed to its super-peer;
• When the query reaches the super-peer, it will be first mapped into a high dimensional point in global index space, followed by KNN search on the global index. By this step, the query will be forwarded $K_{\text{group}}$ most relevant groups;
• Next, by Group index, the query will be further forwarded to $K_{\text{peer}}$ most relevant peers;
• And Finally, by local index, $K_{\text{doc}}$ most relevant documents are returned to the query initiator.
Monitoring XML Data on the Web

The Approximate XML Joins

Outlier Detection for High Dimensional Data

Document Summarization in Information Retrieval

Title Language Model for Information Retrieval

Document Clustering with Cluster Refinement

Document Clustering with Committees

High Dimensional Indexing Using Sampling

Document Summarization
 Updating

• We use a metric AIR (Accumulated Information Ratio) to measure whether rebuilding & indexing summaries is needed.

\[
AIR(dic, dic_{future}) = \frac{\sum_{i=0}^{D[dic_{future}]} |dic[i] - dic_{future}[i]|}{\sum_{i=0}^{D[dic_{future}]} dic[i]}
\]

• AIR represents the changes in peer, group, and global level so far. Only the changes arrive at a certain level severely affecting the system precision, would the rebuilding & indexing be necessary.
Experiments

- We evaluate our system both in a real setting and via simulation;
- Experiment setup

<table>
<thead>
<tr>
<th>Name</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network Type</td>
<td>Power-Law</td>
<td>Topology of network, with outdegree 3.2</td>
</tr>
<tr>
<td>Max_User_Wait_Time</td>
<td>60s</td>
<td>Time for a user to wait an answer</td>
</tr>
<tr>
<td>Query Rate</td>
<td>8e-3</td>
<td>The expected number of queries per user per second</td>
</tr>
<tr>
<td>TTL</td>
<td>5</td>
<td>Time-To-Live of an message</td>
</tr>
<tr>
<td>Network_Size</td>
<td></td>
<td>Number of peers in the network</td>
</tr>
<tr>
<td>Peer_Group_Size</td>
<td></td>
<td>Number of peers in each peer group</td>
</tr>
<tr>
<td>$K_{group}$</td>
<td></td>
<td>Number of super peers to return</td>
</tr>
<tr>
<td>$K_{peer}$</td>
<td></td>
<td>Number of peers for a super peer to return</td>
</tr>
<tr>
<td>$K_{doc}$</td>
<td></td>
<td>Number of documents for a peer to return</td>
</tr>
</tbody>
</table>

TABLE II
Parameters and settings.
Three performance metrics we are interested: Precision of results, Query Response time, and Load (the number of messages being processed).
Retrieval Precision

Implement a relatively small real network: 4 benchmark collections of documents, 30 nodes, with each having around 200 documents. Nodes are clustered into 6 groups, one node in each group is appointed as a super-peer randomly.

- Effect of Dimensionality
  - Three levels (MED dataset);
  - The overall system.
The Effect of Dimensionality (three levels) on Retrieval Precision

Fig. Unit Level

Fig. Peer Level

Fig. Super Level
Precision of the whole system
Retrieval Efficiency

- A simulator with 10,000 peers, each having an average of 2000 synthetic documents; Only results from the first 1000 queries are considered, though queries themselves are generated continuously and endlessly for better simulation.
- More focus are put on studying what factors are involved in a super-peer setting, which may potentially affect the retrieval efficiency.
  - The effect of peer group size on Query Response Time, given a certain query scheduling rate;
  - The relationship between peer group size and the system load;
  - The role of super peer capability in the retrieval efficiency, when the peer group size is increased;
Fig. The effect of Peer Group Size on Query Response Time.
<table>
<thead>
<tr>
<th></th>
<th>size=200</th>
<th>size=400</th>
<th>size=600</th>
<th>size=800</th>
<th>size=1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>L1</td>
<td>227419</td>
<td>149028</td>
<td>123218</td>
<td>108821</td>
<td>99542</td>
</tr>
<tr>
<td>L2</td>
<td>575</td>
<td>850</td>
<td>1092</td>
<td>1309</td>
<td>1597</td>
</tr>
<tr>
<td>L3</td>
<td>0.015</td>
<td>0.038</td>
<td>0.071</td>
<td>0.116</td>
<td>0.225</td>
</tr>
</tbody>
</table>

**TABLE IV**

**System Load (the capability of super peer is same as peer’s)**

<table>
<thead>
<tr>
<th></th>
<th>size=200</th>
<th>size=400</th>
<th>size=600</th>
<th>size=800</th>
<th>size=1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>L1</td>
<td>324110</td>
<td>323848</td>
<td>321272</td>
<td>274668</td>
<td>235062</td>
</tr>
<tr>
<td>L2</td>
<td>722</td>
<td>1419</td>
<td>2056</td>
<td>2436</td>
<td>2884</td>
</tr>
<tr>
<td>L3</td>
<td>5.517e-6</td>
<td>3.31e-5</td>
<td>8.113e-5</td>
<td>0.0003</td>
<td>0.0009</td>
</tr>
</tbody>
</table>

**TABLE V**

**System Load (the capability of super peer is improved by 5)**

L1 – Total Messages transmitted over the network;  
L2 – Average Messages received by a super-peer;  
L3 – Average message queue length of a super-peer, in per 20ms;
Fig. The effect of super peer capability on search (peer group size=400)
Updating Effect

Join effect on Precision
Cost Reduction By Sampling

**Sampling effect on Precision**

![Graph showing sampling rate vs. relative precision]

**Sampling effect on Efficiency**

![Graph showing sampling rate vs. relative cost]
Summary of SummaryIndex

- Hierarchical Summary/Index structure suitable for P2P networks with SuperNodes;
- It can support content-based search efficiently;
- Sampling helps to reduce the cost with retrieval precision unaffected much.
Structured P2P Systems

• DHT-based
  • Chord / Pastry / Tapestry: hash-based into single dimensional space
  • CAN: hash-based into multi-dimensional space
  • P-grid: hash-based into virtual binary search tree

• Skip-list based
  • Skipgraph / SkipNet
• Distributed Hash Table
• $p = \text{hash}(\text{peer})$ and $k = \text{hash}(\text{data item})$
• $p$ and $k$ are uniformly distributed in the same ID space.
• $\text{predecessor}(p)$: 1st node that located anti-clockwise from $p$ on the ID space.
• $\text{successor}(p)$: 1st node that located clockwise from $p$ on the ID space.
• Peer $p$ is responsible to store all objects $k$ such that $k \in [\text{predecessor}(p), p]$
• Routing finger table,
  $\text{pointers of } p_i = \text{successor}(p + 2^{i-1})$
Routing in Chord

- Use fingers (the first finger is its direct successor);
- Route via binary search; always go for the largest predecessor(targetnode);
- Cost is at O(log N);

<table>
<thead>
<tr>
<th>i</th>
<th>succ.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td>2</td>
<td>9</td>
</tr>
<tr>
<td>3</td>
<td>11</td>
</tr>
<tr>
<td>4</td>
<td>14</td>
</tr>
</tbody>
</table>
Chord

- Overlayed $2^m$-Gons
Routing in Chord

- At most one of each Gon
- E.g. 1-to-0
Routing in Chord

Diameter: $\log n$ (1 hop per gon type)
Degree: $\log n$ (one outlink per gon type)
Skip List

- Skip nodes may be random in skip lists.
- Search is $O(\log N)$
Skip Graph

- **Skip List**: A randomized variant of a linked list with additional, parallel lists
- **Skip Graph**: Generalize skip list to provide fault tolerance for distributed environments with more linked lists
  - At each level (logN levels all together), there are multiple linked lists at each level;
  - The bottom level is a doubly-linked list of all nodes in increasing order;
  - Which lists a node belongs to is decided by the node’s *membership vector*, which is generated randomly.
An Example of Skip Graph
Chord Ring = Skip Graph ?
Corner Stitching

Grid-File

- Based on extendible hashing
- Design principle: any point query can be answered in at most 2 disk accesses.
- Two structures: k-dimensional array and k 1-dimensional array
Grid-file

Bucket Overflowed
Grid File
Kd-tree

d-dimensional space is partitioned in zones (subspace) and each is assigned to a node

- Each node is linked with average 2d neighbors;
- Hash-based for data mapping into d-dimensional coordinate space (not native data space);
Routing in CAN

- Routing in CAN is based on spatial proximity, average path length is $O(N^{1/d})$;
Proposals on Range query support

- MANN (Grid 2003)
  - Based on Chord, With a uniform locality preserving hashing;
  - Assume data distribution could be known beforehand;
  - Multi-attribute range queries were supported based on single-attribute resolution;
  - Based on CAN;
  - the inverse Hilbert mapping was used to map one dimensional data space to CAN’s d-dimensional Cartesian space;
- Squid (HPDC 2003)
  - Based on Chord;
  - Hilbert mapping was used;
ZNet

- A distributed system to support multi-dimensional range queries in P2P networks;

- Main features:
  - *The native data space is directly partitioned and indexed, in a way as generalized quad-trees;*
  - *Load balancing is achieved by further partitioning subspaces which may be dense;*
  - *Efficient searching is supported by ordering subspaces with Z-curves at different granularity levels.*
In ZNet, for each peer, besides its object database which stores data objects to be published to the network, it also maintains a virtual database, which contains indices for data objects whose points are covered by the subspace which is managed by the peer.
Adaptive Space Partitioning I

- The space is partitioned in a way as in generalized quad-trees, that is, partitioning occurs along all dimensions at each time.
- Z-curves are used to manage subspaces at different levels;
  - Zones generated by one partitioning are at the same Z-level and form into one scope;
  - Each zone corresponds to a Z-value at a Z-level, and has a unique Z-address;