Routing Indices For Peer-to-Peer Systems

A. Crespo and H. Garcia-Molina


Ioana Varsandan
Jacobs University Bremen
Spring Semester 2007
Outline

• Introduction and Related Work
• Query Processing in a Distributed-Search P2P System
• The Compound Routing Indices (CRI)
• Hop-Count Routing Indices (HRI)
• Exponentially Aggregated RI (ERI)
• Cycles in the P2P Network
• Experimental Results
• Conclusion
• References
Introduction and Related Work

• Search in a p2p system:
  – Structured Network (topology is tightly controlled and documents placed at specified locations):
    • Use Distributed Hash Tables (CAN, CHORD, Pastry, Tapestry)
  – Unstructured Network:
    • Mechanism without an index (Gnutella) flood the entire or part of the network
    • Centralized search systems use specialized nodes which keep an index of the available documents (Napster)
    • Distributed index mechanisms maintain an index at each node.
Introduction – cont.

• This paper discusses a distributed index search mechanism and the use of Routing Indices (RI).

• RIs “suggest a direction” towards the searched document.
  – By using routes instead of destinations, the index size is proportional to the number of neighbors rather than the number of documents.

• Three types of routing indices will be presented:
  – The compound RI
  – The hop count RI
  – The exponential RI
Query Processing in a Distributed-Search P2P System

- Users submit queries along with a stop condition (for ex. the desired number of results)

- When a node receives the query, it first evaluates against its own database (using its local index) and, assuming the stop condition hasn’t been reached, it forwards the query to a subset of its neighbors (typically the ones with most documents)

- If a leaf node is reached, the query is sent back and other neighbors are checked.

- Queries can be forwarded
  - in parallel (better response times, but higher traffic)
  - Sequentially (the focus of this paper)
The Compound Routing Indices (CRI)

- Objective: allow a node to select the best neighbors to forward the query to.
- It is a data structure containing
  - The nr. of documents along each path
  - The nr. of documents on each topic of interest

![Figure 1. A Sample Compound RI](image)
Over- and Undercounts

• For reasons of efficiency, the RI is usually coarser than the local index.

• By keeping a summary of the detailed local index they achieve a more compact RI at the cost of introducing some “errors”:
  
  – A summarization that includes several topics into one may introduce overcounts

  – If the summarization uses a frequency threshold (to minimize update costs) it could also introduce undercounts
The “goodness” heuristic

- Measure of “goodness”: number of documents that can be found along the path.

- Assumptions:
  - Queries are conjunction of subject topics
  - Documents can have more than one topic
  - Topics are independent

- It is equal to: $\text{NrDocs} \times \prod_i (CRI(s_i)/\text{NrDocs})$, where $CRI(s_i)$ is the nr. of docs for topic $s_i$ for a neighbor row

- Example: If A receives a query on topics DB and L, the “goodness” for B would be $100 \times (20/100) \times (30/100) = 6$, for C it’s 0 and for D it’s 75, so node D will be chosen
Using RIs

- The RI consists of:
  - First row is a summary of the local documents
  - Next rows are the CRI

- When A receives a query it first searches its local index, returns pointers to any results found and than computes the “goodness” for each neighbor and forwards to the best node;

![Figure 2. Routing Indices](image)
Creating and Maintaining RIs

- When a new link is formed the 2 nodes exchange aggregates of their RI’s and propagate the information to their neighbors.

- When a RI is updated, the node sends an aggregate of its new RI to its neighbors and the information is further propagated.
  - For efficiency, one can batch multiple updates together and/or ignore small changes.
  - When a node disconnects, its participation is not required, because its neighbors will detect its absence and update their RIs accordingly.

Figure 3: Creating a Routing Index
Hop-Count Routing Indices (HRI)

- Disadvantage of CRI: does not take into account the number of hops required to find a document.

- In the Hop-Count RI, aggregated RI's are stored for each hop up to a "horizon"

```
+----+----+----+----+----+----+----+----+
|    | #DB| N  | T  | L  | #DB| N  | T  | L  |
|----+----+----+----+----+----+----+----+----|
| X  |  60 | 13 |  2 |  5 | 10 |  20|  10|  10|  17|
| Y  |  30 |  0 |  3 | 15 | 12 |  50|  31 | 15 | 20 |
| Z  |   5 |  2 |  0 |  3 |  3 |  70|  10 | 40 | 50 |
```

Figure 4 A sample Hop-count RI for node W
The “goodness” for the HRI

• Cost: nr. of messages

• Goodness: ratio between nr. of docs available through a neighbor and the nr. of messages required to reach them

• This ratio is computed with the help of the regular tree cost model (assumes the network is a regular tree with fanout F)

• Goodness(N_{i,j},q)=\sum_{j=0}^{h}(\text{goodness}(N_{i,j}(j),q)/F^j), where N_{i,j}(j) is the RI entry for j hops through neighbor i;

• Example (for F=3):
  – Goodness(X)=13+10/3=16.33 for a search about DB
Exponentially Aggregated RI (ERI)

• Disadvantages of HRI:
  – no information beyond the horizon
  – Higher storage and transmission costs
• The ERI overcomes all these
  – It stores the result of applying the regular tree cost formula to a hop count RI.
  – Goodness(N)=$\sum_{j=1..th}(\text{goodness}(N(j),T)/F^{j-1})$, where $N(j)$ is the summary of the local index of neighbor j of N, T is the topic, and $th$ is the regular tree height

<table>
<thead>
<tr>
<th>Path</th>
<th>#</th>
<th>DB</th>
<th>N</th>
<th>T</th>
<th>L</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>66.67</td>
<td>16.33</td>
<td>5.33</td>
<td>6.33</td>
<td>15.67</td>
</tr>
<tr>
<td>Y</td>
<td>46.67</td>
<td>10.33</td>
<td>3.00</td>
<td>20.00</td>
<td>18.67</td>
</tr>
<tr>
<td>Z</td>
<td>28.33</td>
<td>5.33</td>
<td>13.33</td>
<td>9.67</td>
<td>19.67</td>
</tr>
</tbody>
</table>

Figure 5. A sample Exponential Routing Index for Node W
Cycles in the P2P Network

- Solutions:
  - No operation (for HRI and ERI, will stop when horizon is reached or due to exponential decay)
  - Cycle avoidance
  - Cycle detection and recovery
    - Have unique identifiers for the originating nodes
Experimental Results

- Cost: Number of messages generated
- Simulator: 95% confidence, <10% relative error
- Compare: CRI, HRI, ERI, No-RI (choose neighbors at random)
- Distribution of documents: uniform and 80/20 biased
  - CRI performs best because it uses the most nodes to compute the goodness of a path

![Comparison of CRI, HRI, and ERI](image)

**Figure 17**: Comparison of CRI, HRI, and ERI
Experimental Results – cont.

- Compared to Gnutella, RI’s reduce the nr. of messages by two orders of magnitude (but Gnutella finds all documents)
- The presented algorithms scale linearly with the number of desired results
- Topologies: tree, tree with added links (cycles), power-law
- RI’s perform better in a power-law topology because:
  - They direct queries to highly connected nodes
  - The average path length between nodes is lower than in tree topologies

Figure 8. Network topology
Experimental Results – Cont.

• Updates:
  – CRI performs bad because it updates all nodes, while HRI and ERI only a subset of nodes

Figure 9. Updates and Network Topology
Conclusions

• CRI, HRI, ERI all perform well in query processing (reduce the nr. of messages by half when compared to No-Ri)

• However, updating the CRI is very costly

• Hence, HRI and ERI are the excellent choice as the search mechanism of a P2P system
References