Keyword Searching in Hypercubic Manifolds

Yu-En Lu, Steven Hand, Pietro Lio
University of Cambridge
Computer Laboratory
Motivation

- Unstructured P2P networks such as Guntella evaluates complex queries by flooding the network while nothing can be guaranteed.
- Distributed Hash Tables evaluates simple queries due to hashing whilst guarantee is provided (at least theoretically).
- What if we may cluster similar objects in similar regions of network via only hashing?
  - No preprocessing is needed
  - No global knowledge required, only the hash
  - Plug & play on top of current DHT designs
## Types of Queries

<table>
<thead>
<tr>
<th>Type</th>
<th>Query Example</th>
<th>System(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Exact Query</strong></td>
<td>$K_1 \land K_2 \land K_3$</td>
<td>DHT Systems</td>
</tr>
<tr>
<td></td>
<td>e.g. “Harry Potter V.mpg”</td>
<td></td>
</tr>
<tr>
<td><strong>Range Query</strong></td>
<td>$K_1 \land K_2 \land K_3 \land K_4 \geq 128$</td>
<td>PHT, Mercury, P-Grid etc.</td>
</tr>
<tr>
<td></td>
<td>e.g. “Harry Potter V.mpg AND bit-rate &gt; 128kbps”</td>
<td></td>
</tr>
<tr>
<td><strong>Partial Match Query</strong></td>
<td>$K_1 \land K_2 \lor K_3 \lor K_4$</td>
<td>Qube</td>
</tr>
<tr>
<td></td>
<td>“Harry Potter [III,IV].mpg’”</td>
<td></td>
</tr>
<tr>
<td><strong>Flawed Query</strong></td>
<td>$K_{i+1} \land K_{j-20}$</td>
<td>Qube</td>
</tr>
<tr>
<td></td>
<td>“Hary Porter.mpg”</td>
<td></td>
</tr>
</tbody>
</table>
A Projection of Object Feature Space

Possible answers to the query

Harry

Movie

French Cuisine
Each object is represented as a bit-string where 1 denotes it contains a keyword and 0 means not. Each bit string is then hashed onto the P2P name space. The nodes in the network chooses positions in the P2P space randomly and links with each other in some overlay topology. In our case, a hypercube is used.
Design Principles and Fundamental Trade-offs

- **Latency vs. Message Complexity**
  - Low message complexity usually means low latency
  - Not entirely true for DHT systems

- **Fairness vs. Performance**
  - Sending everything to a handful of ultra-peers is fast and simple
  - Having things spread across the network means fairer system (and perhaps better availability)

- **Storage vs. Synchronisation complexity**
  - Most popular queries may be processed by querying one random node due to generous replication/caching
  - For some applications such as distributed inverted index, frequent synchronisation is costly
Hashing and Network Topology

- **Summary Hash**
  \[ h : \{0, 1\}^* \rightarrow \{0, 1\}^b \]
  - Non-expansive:
    \[ d(x, y) \leq d(h(x), h(y)) \]
  - Fair Partitioning:
    \[ h^{-1}(u) = h^{-1}(v) \]

- **Keyword edges linking nodes in one word distance**
  - Similar objects are located in manifolds of the Hypercube
Query Processing

Harry Potter
Harry Potter Music
Harry
VII
Experimental Setup

- A Hypercube DHT is instantiated where end-to-end distance is drawn from King dataset*
  - King dataset contains latency measurements of a set of DNS servers
  - Surrogates to a logical ID is chosen based on Plaxton style post-fix matching scheme
- Nodes choose DHT IDs randomly
  - No network proximity to expose worst case performance and tradeoff with dimensionality and caching
- A sample of FreeDB**, a free online CD album containing 20 million songs, is used to reflect actual objects in real world
- Gnutella query traces served as our query samples

* http://pdos.csail.mit.edu/p2psim/kingdata/
** http://www.freedb.org
* Recall rate = percentage of relevant objects found

** Legend tuples denotes (b,n) where b is dimensionality and n is the size of the network
Query Latency in Wide Area

Selection of b controls the degree of clustering
Network Performance

* This result takes the query “bon jovi” for example where there are 3242 distinct, related songs in FreeDB
Conclusion

- Qube spreads objects across the network by their similarity
  - Better fairness and availability
  - Zero preprocessing
  - Little synchronisation need
  - By tuning parameter $b$, one may choose the degree of performance/fairness tradeoff

- Further investigating lower latency schemes to trim probing cost and decouple query accuracy from network size
Future Work

- Large scale simulation (>100K nodes with realistic network latency generator)
- Flash crowds query model and replication/caching
- Distributed proximity-searches such as kNN under Euclidean metric
Thank you!

Yu-En.Lu@cl.cam.ac.uk