Resource and Query Aware, P2P-Based Multi-Attribute Resource Discovery

H. M. N. Dilum Bandara and Anura P. Jayasumana

Department of Electrical and Computer Engineering, Colorado State University, Fort Collins, CO 80523, USA. 
{dilumb, anura}@engr.ColoState.edu

CASA is primarily supported by the Engineering Research Centers Program of the National Science Foundation under NSF award number 0313747.
Abstract

Distributed, multi-attribute Resource Discovery (RD) is a fundamental requirement in collaborative P2P, grid, & cloud computing. We present an efficient & load balanced, P2P-based multi-attribute RD solution that consists of 5 heuristics, which can be executed independently & distributedly. 1st heuristic maintains a minimum number of nodes in a ring-like overlay consequently reducing the cost of resolving range queries. 2nd & 3rd heuristics dynamically balance the moderate key & query load by transferring keys to neighbors & by adding new neighbors when existing ones are insufficient. Last 2 heuristics, namely fragmentation & replication, form cliques of nodes that are placed orthogonal to the overlay ring to dynamically balance the highly skewed key & query loads while reducing the query cost. By applying these heuristics in the presented order, a RD solution that better responds to real-world resource & query characteristics is developed.
Problem Statement

- **Overlay ring-based resource discovery**
  - Pros – Scalable & performance guarantees
  - Cons – High query \(O(N)\) & advertising cost, & unbalanced load
    - Conventional solutions – Domain of attributes \(D_i \gg N\)

- **Real-world resources & queries [1-3]**
  - Domain of some attributes is small \(D_i \ll N\)
  - Queries with large range of attribute values
    - Not useful to advertise even attributes with large \(D_i\) at high resolutions
    - Effectively, \(D_i \ll N\)

- **Problem**

  \[
  \text{minimize } N \\
  \text{subject to } I^r_{Cap}, Q^r_{Cap}
  \]

\(N\) – No of nodes \(I^r_{Cap}\) – Index capacity of a node \(Q^r_{Cap}\) – Query capacity of a node
System Model

$Q_{i - 1}^{Fwd}$ – Queries forwarded from predecessor $i - 1$

$Q_{In}^i$ – Queries that start query resolution at node $i$

$Q_{Out}^i$ – Queries that go out of overlay at node $i$

$Q_{Fwd}^i$ – Queries forwarded from predecessor $i - 1$
Heuristic 1 – Prune Nodes With Lower Contribution

Remove nodes with lower contribution to query resolution

a) Remove $c \rightarrow$ Reduce query cost $Q_{Out}^c = 0$
   - Can $b$ or $d$ accept any resources indexed at $c$?
   - $d$ is preferred as no changes are required to overlay network

b) Remove $a$, $b$, or $d \rightarrow$ Reduce query cost $Q_{Out}^i < Q_{Thr}^i$
   - Can neighbors accept resource index & query load?
   - Successor is preferred
Heuristics 2 & 3 – Key Transfer

When nodes are already contributing but overloaded

• **Heuristic 2**
  – Node $i$ is overloaded
  – Adjust address range
    • Move keys/resources to successor or predecessor
    • Can it accept?
  – Successor is preferred
  – Minor changes to overlay

• **Heuristic 3**
  – Node $i$ is overloaded & successor & predecessor not willing to accept load
  – Add new successor or predecessor
    • Load must not exceed capacity of a node
  – Successor is preferred
  – Some changes to overlay
• Heuristics 2 & 3 will fail if load is too much for a node

In practice, nodes can index many resources & answer many queries/second → Cliques are not large
Simulation Setup

- Compared with a Chord network with same number of nodes
  - Our solution is always better than any solution that add all the nodes to overlay
- Workloads derived from real-world resource & query traces

<table>
<thead>
<tr>
<th>Workload</th>
<th>Resources</th>
<th>Queries</th>
</tr>
</thead>
<tbody>
<tr>
<td>File sharing</td>
<td>100,000 copies of 10,000 distinct files, ~Zipf’s(0.7).</td>
<td>Case 1 – ~Zipf’s(0.5), Case 2 – ~Zipf’s(1.0). Query arrival ~exponential(2 min).</td>
</tr>
<tr>
<td>CPU speed</td>
<td>CPU speed of 100,000 randomly sampled nodes from SETI@home. Can be approximated by ~N(2.36, 0.28) [1, 3].</td>
<td>Pulse-like queries derived from PlanetLab. Use empirical CDF to generate ranges of attribute values. Query arrival ~exponential(2 min).</td>
</tr>
<tr>
<td>CPU free</td>
<td>A synthetic dataset of 100,000 CPU free values derived using linearly-interpolated empirical CDF from PlanetLab.</td>
<td>Pulse-like queries derived from PlanetLab. Use empirical CDF to generate range of attribute values. Query arrival ~exponential(2 min).</td>
</tr>
<tr>
<td>PlanetLab</td>
<td>527 node PlanetLab trace with 12 static &amp; 12 dynamic attributes. Also used 250, 750, &amp; 1000 node traces generated using [5-6].</td>
<td>Synthetic trace generated using empirical CDFs derived from set of attributes in a query, their popularity, [l_i, u_i], &amp; m [7]. ~exponential(10 sec).</td>
</tr>
</tbody>
</table>
Performance Analysis – Single-Attribute

Query load distribution of file sharing workloads at steady state

Load distribution of CPU free workload

Ave. hop count required to resolve queries at steady state
Each heuristic addresses a specific problem

More efficient & load balanced solution when all 5 heuristics are combined

- Work with both single & multiple attributes
Summary & Future Work

• 5 heuristics for
  – Efficient P2P-based multi-attribute resource discovery
  – Better load distribution & meet node capacity constraints

• Heuristics rely on local statistics to capture complex characteristics of real-world resources & queries
  – Support both single-attribute & multi-attribute resources

• Currently extending solution to
  – Also balance load due to
    • Frequent advertising of dynamic resources
    • Messages forwarded by overlay nodes
  – Support resource matching & binding
Related Publications


www.cnrl.colostate.edu/Projects/CP2P/