A Hierarchical Semantic Overlay for P2P Search

Tao Gu*, Hung Keng Pung, Daqing Zhang

*Research Scientist, Institute for Infocomm Research
*Email: tgu@i2r.a-star.edu.sg
*URL: www1.i2r.a-star.edu.sg/~tgu
Outline

• Motivation
• Our approach
  – Overview
  – Data model
  – Ontology design
  – Semantic clustering
  – Peer identification
  – Top-level overlay
  – Low-level overlay
• Some preliminary results
• Conclusion
Motivation

- **Unstructured** P2P systems
  - Pros: do not impose any structure on the data; easy to handle the dynamic changes of peers and their data; low overlay maintenance cost, etc.
  - Cons: flooding-based routing algorithm generates large amount of redundant messages; not scalable.

- **Structured** P2P systems
  - Pros: efficient routing; good scalability, etc.
  - Cons: data placement and network topology are tightly controlled; high overlay maintenance cost.

- **Hybrid** P2P systems
  - Combine the advantages of both *unstructured* and *structured* P2P systems
  - Our approach - *A Hierarchical Semantic Overlay Network* falls in this category
Overview of Our Approach

• Ontology-based two-level semantic overlay
  – Top-level overlay: peers are grouped into a semantic cluster based on ontologies; semantic clusters are organized into a one-dimensional ring space.
  – Low-level overlay: semantic clusters can be organized into unstructured overlay or DHT-based overlay.

• Abstract data semantic based on ontologies
  – Hierarchical design for ontologies

• A DHT-based inter-cluster routing algorithm
Data Model

• The basic model – an RDF triple
  – <subject predicate object>
  – E.g., <socam:TaoGu socam:homeAddress “XYZ”>, or <socam:TaoGu socam:locatedIn socam:LivingRoom>

• Machine-understandable, -processable, good interoperability. Limit to representation methods.
Ontology Design

- Two-level hierarchy in the ontology design
  - The upper ontology defines common concepts in a computing/application domain
  - Lower ontologies define details/own concepts.
- Why two-level hierarchy?
  - A peer defines/stores its own lower ontology based on context data, no need to store all – smaller metadata size.
  - It allows the construction of a semantic P2P overlay network.
Ontology-based Semantic Clustering

• The basic principle:
  – The leaf nodes in the upper ontology are used as semantic clusters.
  – If the predicate of a data triple is of type *ObjectProperty*, we use `<pred obj>` pair
  – If the predicate of a data triple is of type *DatatypeProperty*, we use `<sub pred>` pair

Context data triples:

Eg1: `<socam:John socam:locatedIn socam:Bedroom>`

Eg2: `<socam:Bedroom socam:lightLevel 'LOW'>`

Diagram: [Diagram of context data triples involving IndoorSpace, OutdoorSpace, and conditions]
Peer Identification

- **Semantic Cluster ID**
  - $sid = \text{hash}(\text{"a leaf node in the upper ontology"})$

- **Peer ID**
  - $\text{peer id} = [\text{hash}_m(\text{"a leaf node in the upper ontology"})][\text{hash}_n(\text{"IP address"})]$
Top-level Overlay

semantic clusters
(leaf nodes in the upper ontology)

A unstructured low-level overlay with cluster splitting/merging and parallel flooding

$q \leftarrow q \ (query)
A Chord-based Low-level Overlay

By <pred obj> pair
<socam:TaoGu socam:homeAddress “XYZ”>

By <sub pred> pair
<socam:TaoGu socam:homeAddress “XYZ”>

By <sub obj> pair
<socam:TaoGu “XYZ”>

Pairs: <sub pred> or <pred obj> or <sub obj>
<table>
<thead>
<tr>
<th>Hash value in [0,7]</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;socam:TaoGu socam:homeAddress&gt; 6</td>
</tr>
<tr>
<td>&lt;socam:homeAddress “XYZ”&gt; 1</td>
</tr>
<tr>
<td>&lt;socam:TaoGu “XYZ”&gt; 4</td>
</tr>
</tbody>
</table>
Some Preliminary Results

• Simulation Setup
  – Two types of network topologies in our model: physical topology and P2P overlay topology.
  – Parameters: \( m \) – number of bits to represent semantic cluster, \( n \) – number of bits to represent sub-cluster, \( M \) – cluster size, \( N \) – network size

• Performance metrics
  – Fraction of nodes contacted per query
  – Search path length
  – Search cost
  – Maintenance cost
Search Path Length

- The average number of hops traversed by a query to the destination.
- \( N = 2^8 \sim 2^{13} \)
- \( M = 1 \) (disable clustering effect)
- \( n = 0 \) (disable parallel search)
- \( \beta = 1/4 \) or \( 1/2 \) or \( 1/2^m \)

![Graph showing search path length vs. number of nodes for different configurations of Gnutella, SONs, and SCS with \( \beta = 1/2^m \), \( \beta = 0.25 \), and \( \beta = 0.5 \).]
Search Cost

• The average number of query messages incurred during a search operation in the network.
• $N$ from $2^8$ to $2^{13}$
• $m = 5$
• $n = 0$ or $2, 3$
• $\beta = 1/4$ or $1/2$ or $1/2^m$
Maintenance Cost

- The average number of messages incurred when a node joins or leaves the network. It consists of the costs of node joining and leaving, cluster splitting and merging, and index publishing.

- $M = 32$
- $n = 2$
- $m = 1 \sim 8$
- $\beta = 1/4$ or $1/2$ or $1/2^m$
Conclusion

• Conclusion
  – A hybrid approach to P2P search
  – Preliminary results shows efficiency
• On-going work
  – Building the simulator for the chord-based low-level overlay
  – Further evaluate the performance