TOPLUS

Topology-centric Look-Up Service

Luis Garcés-Erice
Keith W. Ross
Ernst W. Biersack
Pascal A. Felber
Guillaume Urvoy-Keller

Institut Eurécom – Sophia Antipolis, France
Outline

Very brief introduction to DHTs
Motivation and Goals
TOPLUS
Benchmarks
Conclusion
Introduction to DHTs

P2P Lookup Services

Assign resources to peers
Locate resources upon request

Structured P2P systems:

A key is a resource identifier.

A Hash Table maps a key to a bucket through a hash function $h()$

$$h(key) \rightarrow 0,1,2 \ldots N-1$$, all N possible outputs of $h()$

In Distributed Hash Tables (DHTs) each bucket is a peer

Key assigned to peer with the “closest” id

Chord, CAN, Pastry

Scalability: Each peer knows a subset of all peers

$h(key)$ must be routed to corresponding “bucket”
Motivation

P2P DHT overlay networks are built on top of the Internet

Queries are routed through the overlay

Neighbours in the overlay network are not close in the physical Internet network

Pastry uses closest neighbour to route (\textit{closest} $\neq$ \textit{close})
Goal

Overlay Goal

- Provide small stretch: route queries in the overlay as close as possible to IP path

How can the lookup path “follow” the IP path?

- Want to get close to the destination in the first step
- Stretch close to 1

Our solution

**TOPLUS** (TOPology-centric Look-Up Service)

- A “benchmark” to topology-aware DHTs
Node ID = IP@ of node

IP Prefix ranges define (nested) groups

Every node needs to know a “delegate” in (some of) the other groups

TOPLUS - TOPology-centric Look-Up Service

DHT LookUp delay
IP Routing delay
Node State

More detail about closer groups
Coarser knowledge about groups far away

\[ H_0 = I \]

IP Addresses
Key Look-Up (I)

Node \( n=1.2.3.4 \) looks-up key \( k=193.56.1.2 \)

Number of hops \( \leq H+1, \quad H = \text{height of tree} \)
Key Look-Up (II)

Each key $k$ is a bit string of length $m > 32$:

First 32-bits used for routing to node responsible for that key

XOR metric:

Let node $j = j_{31}j_{30}...j_0$ and $k = k_{31}k_{30}...k_0$: $d(j, k) = \sum_{i=0}^{31} \left| j_i - k_i \right| \cdot 2^i$

Note that closest ID is unique:

$$d(j, k) = d(j', k) \iff j = j'$$

Refinement of longest-prefix match

Example (8 bits)

$$k = 10010110$$

$$j = 10110110 \quad d(j, k) = 2^5 = 32$$

$$j' = 10001001 \quad d(j', k) = 2^4 + 2^3 + 2^2 + 2^1 + 2^0 = 31$$
Key Look-Up (III)

XOR metric solves “black hole” problem

Example (32 bit):

Key $k = 11100011110101001011100101011001$
Node $j = 11100011110101001011100101011000$
Node $j' = 11100011110101001011100101011101$

Simple longest-prefix matching cannot decide
However $d(j,k) = 101 > d(j',k) = 100$
node $j'$ responsible for key $k$
Overlay Maintenance

A new node $n$ joining the TOPLUS network:

Delegate diversity property
Robust links between groups, no DHT reconstruction when node leaves
On-Demand Caching

\[ k = 212.17.89.100 \quad 193.21/16 \quad k' = 193.21.89.100 \]
Where to get the IP prefixes from?

Use to create the TOPLUS hierarchy:

We obtain **250,000** IP prefixes from

**BGP tables** (Oregon U., Michigan U. (Merit Network))

**Routing registries** (RIPE, Castify Networks)
Partial Order Tree Construction (I)

Very wide; very unbalanced

+ 47,000 tier-1 groups

Small groups on tier-1 (long prefixes e.g. 121.45.62.128/27)

Deep hierarchies

Huge routing table!
**Partial Order Tree Construction (II)**

**Modified** trees:
- Reduce routing table size
- Respect topology as possible

Create tier-1 virtual groups
- 152.67.29/24
- 152.8.121.128/26
- ... 152/8
- 152.212.107/28

Add a level to the hierarchy
Tested Trees

Original: direct hierarchy obtained from IP prefixes

8-bit: Only prefixes from 8 to 16 bit on tier-1

152.153/12
152.53.128/24
152.53.121.128/26
...  
152.53.107.0/28

152/8 

152.153/12 does not get aggregated

Original+1: Allow only 8-bit prefix on tier-1

152.153/12
152.53.128/24
152.53.121.128/26
...  
152.53.107.0/28

152/8 

152.153/12 gets aggregated

3-Tier: three fixed tiers, 8-, 16- and 24-bit
Benchmarking: Stretch
King

From University of Washington, Seattle
Given 2 IP addresses, measures distances between corresponding DNS servers
It can at least give an **estimate** of delay between two nodes
Benchmarking: Measurements

Stretch:
- Measured from a node in our site to 1,000 random valid destinations

Routing table size:
- Averaged over 5,000 random nodes

Observe DHT trade-off:
- Stretch vs. Routing Table size
**Benchmarking: Stretch (II)**

Average Stretch **per tier**: The deeper in the hierarchy, the larger the stretch

<table>
<thead>
<tr>
<th>Tier</th>
<th>Original</th>
<th>8-bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>2</td>
<td>1.29</td>
<td>1.53</td>
</tr>
<tr>
<td>3</td>
<td>1.31</td>
<td>1.56</td>
</tr>
<tr>
<td>4</td>
<td>1.57</td>
<td>1.58</td>
</tr>
</tbody>
</table>
## Benchmarking: Routing Table

Routing Table size: Average of 5,000 random peers

<table>
<thead>
<tr>
<th>Tier</th>
<th>Original</th>
<th>8-bit</th>
<th>Original+1</th>
<th>3-Tier</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>47,467</td>
<td>8,593</td>
<td>143</td>
<td>143</td>
</tr>
<tr>
<td>2</td>
<td>47,565</td>
<td>8,713</td>
<td>436</td>
<td>248</td>
</tr>
<tr>
<td>3</td>
<td>47,654</td>
<td>8,821</td>
<td>831</td>
<td>261</td>
</tr>
<tr>
<td>4</td>
<td>47,796</td>
<td>8,950</td>
<td>1,279</td>
<td>-</td>
</tr>
<tr>
<td>5</td>
<td>47,890</td>
<td>9,016</td>
<td>696</td>
<td>-</td>
</tr>
</tbody>
</table>

- **Very large tier-1**
- **8-bit Prefix tier-1**
## Benchmarking: Stretch vs R. Table Size

Inversely proportional

<table>
<thead>
<tr>
<th></th>
<th>Stretch</th>
<th>RT Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original</td>
<td>1.17</td>
<td>47,547</td>
</tr>
<tr>
<td>8-bit</td>
<td>1.28</td>
<td>8,699</td>
</tr>
<tr>
<td>Original+1</td>
<td>1.90</td>
<td>656</td>
</tr>
<tr>
<td>3-Tier</td>
<td>2.32</td>
<td>211</td>
</tr>
</tbody>
</table>
Conclusion

Topology-centric DHT design

Features
- Small stretch
- Fast XOR-based routing
- Natural On-demand P2P caching hierarchy
- Static deployment very easy

Open Issues
- Non-uniform population of ID space
- Correlated node failures