Pastry

Peter Druschel, Rice University
Antony Rowstron, Microsoft Research UK

Some slides are taken from the authors original presentation
What is Pastry?

Pastry is a structured P2P network
What is Pastry

• Self-organizing overlay network
• Lookup/insert object in \( < \log_{16} N \) routing steps (expected)
• \( O(log N) \) per-node state (for routing table)
• Network proximity routing
Pastry: Object distribution

Consistent hashing
[Karger et al. ‘97]

128 bit circular id space

nodelds (uniform random)

objld (uniform random)

Invariant: node with numerically closest nodeld maintains object
Pastry: Routing

Routing table for node \(65a1fc\) \((b=4, \text{so } 2^b = 16)\)
Pastry Node State

State of node **10233102**

<table>
<thead>
<tr>
<th>Leaf set</th>
<th>SMALLER</th>
<th>LARGER</th>
</tr>
</thead>
<tbody>
<tr>
<td>10233033</td>
<td>10233021</td>
<td>10233120</td>
</tr>
<tr>
<td>10233001</td>
<td>10233000</td>
<td>10233230</td>
</tr>
</tbody>
</table>

Set of nodes with \(|L|/2\) smaller and \(|L|/2\) larger numerically closest NodeIds

<table>
<thead>
<tr>
<th>Routing table</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>-0-2212102</td>
<td>1</td>
<td>-2-2301203</td>
<td>-3-1203203</td>
</tr>
<tr>
<td>0</td>
<td>1-1-301233</td>
<td>1-2-230203</td>
<td>1-3-021022</td>
</tr>
<tr>
<td>10-0-31203</td>
<td>10-1-32102</td>
<td>2</td>
<td>10-3-23302</td>
</tr>
<tr>
<td>102-0-0230</td>
<td>102-1-1302</td>
<td>102-2-2302</td>
<td>3</td>
</tr>
<tr>
<td>1023-0-322</td>
<td>1023-1-000</td>
<td>1023-2-121</td>
<td>3</td>
</tr>
<tr>
<td>10233-0-01</td>
<td>1</td>
<td>10233-2-32</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td>102331-2-0</td>
<td></td>
</tr>
</tbody>
</table>

Prefix-based routing entries

<table>
<thead>
<tr>
<th>Neighborhood set</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>13021022</td>
<td>102000230</td>
<td>11301233</td>
<td>31301233</td>
</tr>
<tr>
<td>02212102</td>
<td>22301203</td>
<td>31203203</td>
<td>33213321</td>
</tr>
</tbody>
</table>

|M| “physically” closest nodes
Pastry: Routing

Properties

$\log_{16} N$ steps to search

$O(\log N)$ size of routing table
Pastry: Leaf sets

Each node maintains IP addresses of the nodes with the L/2 numerically closest larger and smaller node IDs, respectively.

- routing efficiency/robustness
- fault detection (keep-alive)
- application-specific local coordination
Pastry: Routing procedure

if (destination is “within range of our leaf set”)  
forward to numerically closest member
else
  let $l =$ length of shared prefix
  let $d =$ value of $l$-th digit in $D$’s address
  if ($R_l^d$ exists) forward to $R_l^d$
    ($R_l^d =$ $l^{th}$ row & $d^{th}$ col of routing table)
  else forward to a known node that
    (a) shares at least as long a prefix, and
    (b) is numerically closer than this node

[Prefix routing]
Pastry: Performance

Integrity of overlay/ message delivery:
• guaranteed unless $L/2$ simultaneous failures of nodes with adjacent nodeIds occur

Number of routing hops:
• No failures: $< \log_{16} N$ expected
• $O(N)$ worst case (why?), average case much better
Pastry: Self-organization

Initializing and maintaining routing tables and leaf sets

• Node addition
• Node departure (failure)
Pastry: Node addition

New node X: d46a1c

The new node X asks node 65a1fc to route a message to it. Nodes in the route share their routing tables with X.
Node departure (failure)

Leaf set members exchange heartbeat messages

- **Leaf set repair (eager):** request set from farthest live node in set
- **Routing table repair (lazy):** get table from peers in the same row, then higher rows
Node departure (failure)

Leaf set members exchange heartbeat

- **Leaf set repair (eager):** request the set from farthest live node
- **Routing table repair (lazy):** get table from peers in the same row, then higher rows
Pastry: Average # of hops

L=16, 100k random queries
Pastry: Proximity routing

Proximity metric = time delay estimated by a ping
A node can probe distance to any other node

Each routing table entry uses a node close to the local node (in the proximity space), among all nodes with the appropriate node Id prefix.
Pastry: Routes in proximity space

Nodelp space

Route(d46a1c)

d46a1c

d471f1
d467c4
d462ba

d13da3

Proximity space

d462ba

d4213f

d467c4

65a1fc

d13da3

65a1fc
Pastry: Distance traveled

L=16, 100k random queries, Euclidean proximity space
PAST: File storage

Storage Invariant:
File “replicas” are stored on $k$ nodes with node IDs closest to fileId

(\(k\) is bounded by the leaf set size)
PAST: File Retrieval

fileId

file located in $\log_{16} N$ steps (expected)

usually locates replica nearest to client C
PAST API

• *Insert* - store replica of a file at $k$ diverse storage nodes
• *Lookup* - retrieve file from a nearby live storage node that holds a copy
• *Reclaim* - free storage associated with a file

Files are *immutable*
SCRIBE: Large-scale, decentralized multicast

- Infrastructure to support **topic-based publish-subscribe** applications
- Scalable: large numbers of topics, subscribers, wide range of subscribers/topic
- Efficient: low delay, low link stress, low node overhead