

---

# Flexible Routing in Grouped DHTs

---

Yiming Zhang<sup>1</sup>

Dongsheng Li<sup>1</sup>

Lei Chen<sup>2</sup>

Xicheng Lu<sup>1</sup>

<sup>1</sup> National University of Defense Technology (China)

<sup>2</sup> Hong Kong University of Science and Technology

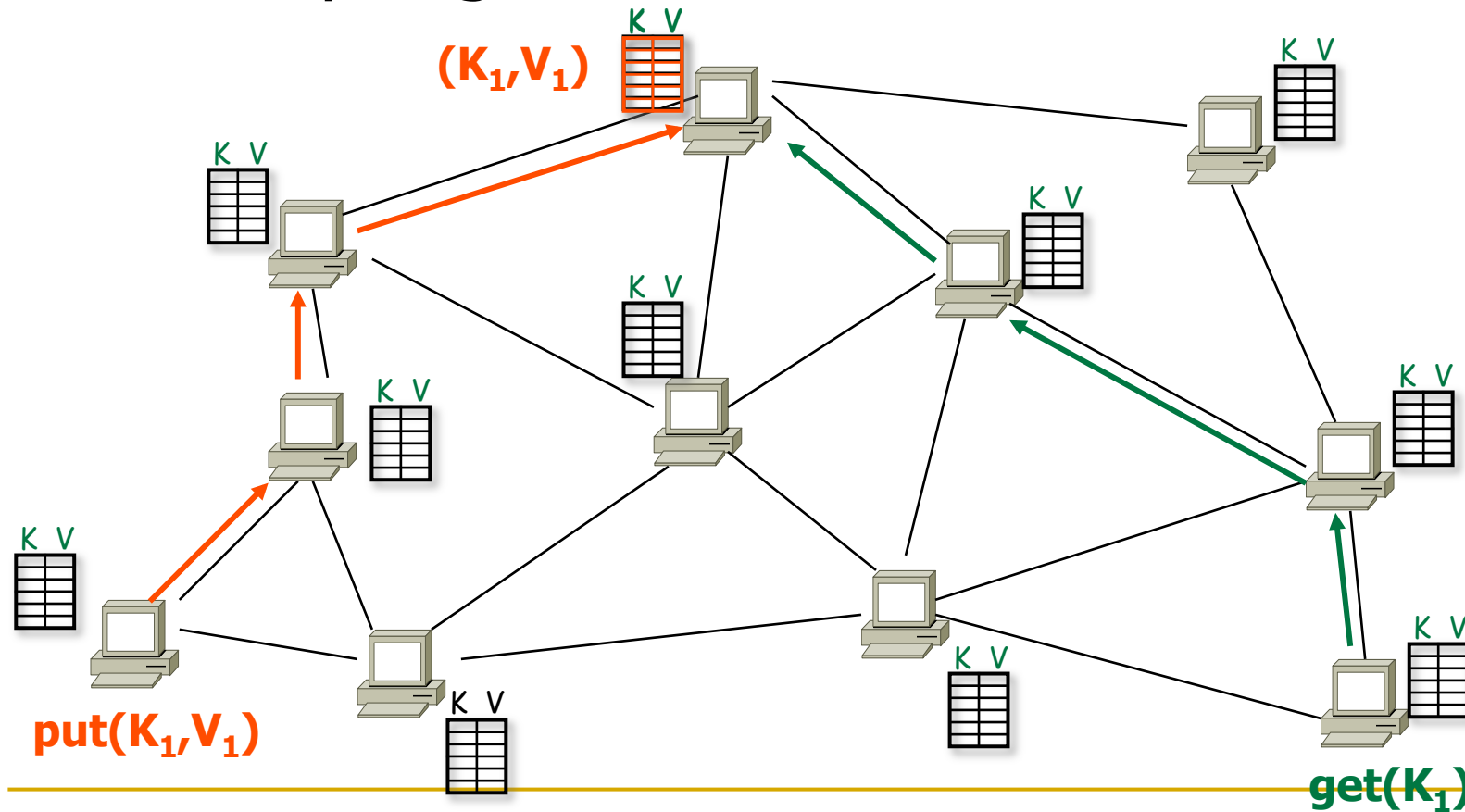
---

# INSPIRATION



# Example of DHTs

- A DHT is just a “distributed” hash table.
- Use the put/get interface to store/fetch data.



---

# DHTs' advantage and price

- DHTs have emerged as a general infrastructure.
- The **advantages** of DHTs
  - There's a uniform distribution of load among all nodes
  - DHTs provide a simple put/get interface.
- What's the **price** of these advantages
  - Treat all nodes equally without discrimination
  - Give up control of
    - Routing destination
    - Routing path
  - All messages are routed using a common algorithm.

---

# DHTs' expectation and reality

- The **ideal** world of DHTs...
  - All nodes are homogeneous
  - No preference for the routing path or destination
- The **real** world...
  - Nodes might be heterogeneous with respect to
    - Capabilities, reputations, affiliations of domains
    - Some nodes can be trusted, while others are malicious
  - Various preferences for message routing
    - Where to safely store the data
    - What kind of nodes should be passed by in the routing

---

# Solution

- Provide the ability to organize nodes into **groups**
  - ❑ Put similar nodes in one group, such as a group of trusted nodes, or a group of fast nodes
  - ❑ Note that we do NOT address how to distinguish the heterogeneity, e.g. identify which are trusted nodes.
- Support **flexible** forms of DHT routing
  - ❑ Destination-Specified (DS) routing
  - ❑ Path-Constrained (PC) routing
  - ❑ Normal DHT routing

# Definition of DS/PC routing

## ■ Notation

- $r(K)$ : the root for  $K$  in the DHT
- $r_X(K)$  the root for  $K$  in group  $X$

## ■ Destination-Specified (DS) routing

- Given a group  $X$  and a message with key  $K$ , DS routing refers to the routing that terminates at  $r_X(K)$

## ■ Path-Constrained (PC) routing

- Given a group  $X$  and a message with key  $K$ , suppose it originates from a group- $X$  node, PC routing refers to the routing in which each hop is constrained within  $X$  and terminates at  $r_X(K)$

---

# DS/PC routing provide benefits.

## ■ Performance

- DS routing can improve performance by allowing one to uniformly distribute a service in the group of qualified nodes
- E.g., a computation-intensive service might wish to limit itself to the group of nodes with fast CPUs.

## ■ Availability

- DS routing can improve availability by specifying the routing destination to a subset of stable nodes
- If nodes are organized into groups in terms of domains, failures outside a group will never affect the PC routing in that group

## ■ Security

- DS routing can improve security by storing data within the group of nodes with high reputation or within a trusted domain
- PC routing provides additional security by guaranteeing the traffic in a group  $X$  will never be exposed to nodes outside  $X$ .

---

# In the following of this talk ...

- Data structures
- Flexible routing
- Support hierarchical groups.

# DATA STRUCTURES (G-TAP)



---

# Challenge and solution

- The challenge we face
  - to support arbitrary group formation
  - to achieve scalable and fast group discovery
- The key idea behind G-TAP
  - Embedding for each group in the base DHT
    - a sub-DHT routing structure
    - a group membership rendezvous (GMR) tree

# Review Tapestry design

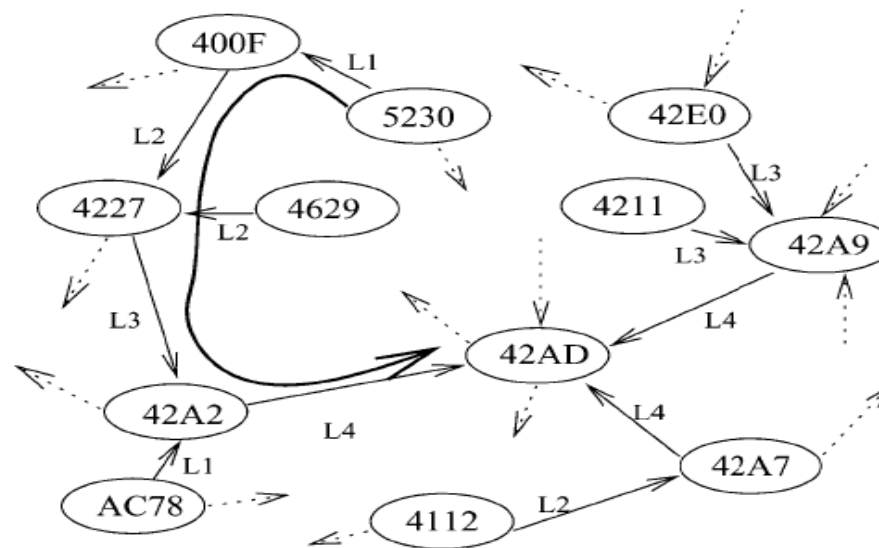
- A Tapestry node has
  - NodeID of  $n$  base- $b$  digits
  - Routing table composed of entry set and leaf set
    - Each entry in the entry set maintains  $c$  links, including one primary link and  $c - 1$  backup links.
    - E.g. entry set of node 1023 with  $n=4$ ,  $b=4$ ,  $c=2$

	Col 0	Col 1	Col 2	Col 3
Row 0				
Row 1				
Row 2				
Row 3				

- Leaf set maintains  $2f$  leaves numerically closest to the node

# Review Tapestry design (cont.)

- All Tapestry nodes form a Plaxton tree routing structure
  - Bit-correct routing algorithm
  - E.g. route from 5230 to 42AD

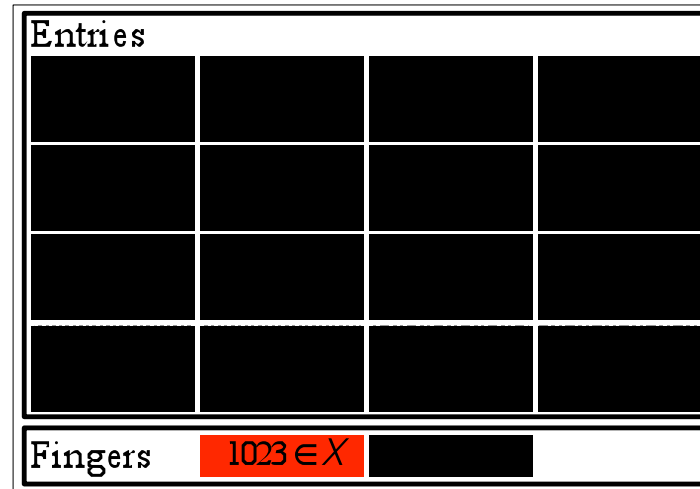


# Entry set & leaf set in G-TAP

- Note that a Tapestry node can often choose among many different links for a given entry
  - Tapestry utilizes this diversity to minimize lookup latency.
- G-TAP utilizes this diversity of neighbor choices to embed sub-DHTs in the overlay
- Suppose node  $u = u_1 u_2 \dots u_n$  belongs to group  $X$ , then each entry  $E_{m,i}$  in node  $u$ 's **entry set** contains
  - A primary link points to  $v = v_1 v_2 \dots v_n$  satisfying
    - $v_j = u_j$  and  $v_{m+1} = i$
    - $v$  should be numerically closest to address  $u_1 u_2 \dots u_m i u_{m+2} \dots u_n$ .
  - A group- $X$  link points to  $w = w_1 w_2 \dots w_n$  satisfying
    - $w_j = u_j$  and  $w_{m+1} = i$
    - $w$  should be a group- $X$  node
  - $c - 2$  backup links

# Entry set & leaf set in G-TAP (cont.)

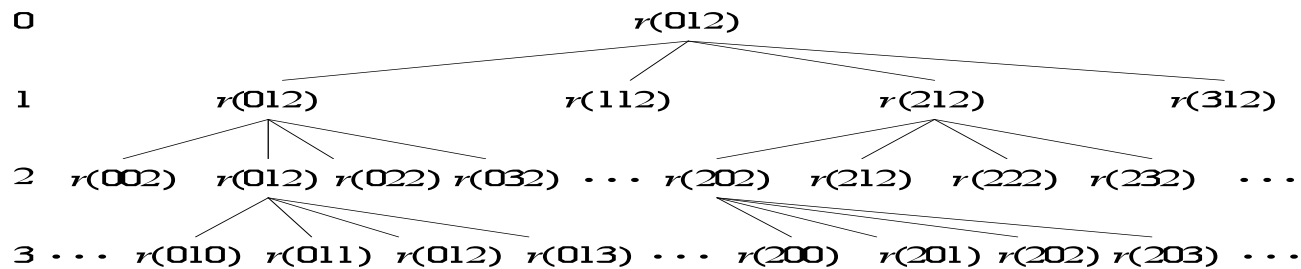
- E.g. Entry set of a group- $X$  node 1023 ( $n=4, b=4, c=2$ )



- Note although this entry set contains no backup link, the group- $X$  link can play exactly the same role as a backup link.
- The **leaf set** of a G-TAP node maintains
  - $2f$  primary leaves for the base DHT, and
  - $2f$  leaves for group  $X$

# Group Membership Rendezvous tree

- GMR tree: to support fast group discovery
  - Query load is hierarchically distributed in a set of rendezvous
- GMR tree for arbitrary Group  $Z$  ( $\text{Hash}(Z)=z_1z_2\dots z_n$ )
  - has  $n+1$  levels
  - at each level  $i$  the  $b^i$  nodes are of the form  $r(x_1\dots x_i z_{i+1}\dots z_n)$
  - each inner node  $r(x_1\dots x_i z_{i+1}\dots z_n)$  at level  $i$  has  $b$  children at level  $i+1$  of the form like  $r(x_1\dots x_i j z_{i+2}\dots z_n)$



- Each node has a finger to any group- $Z$  node in its sub-tree.

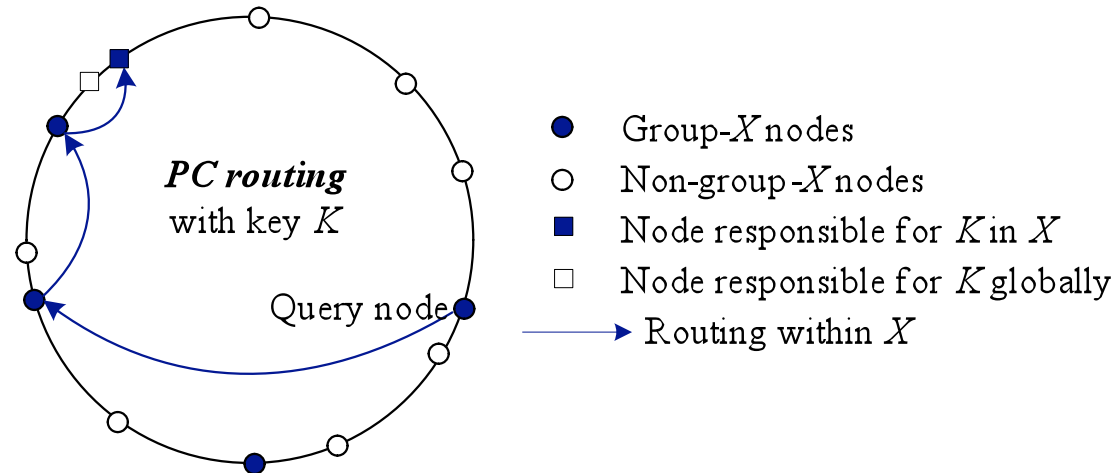
---

# FLEXIBLE ROUTING



# Path-Constrained routing

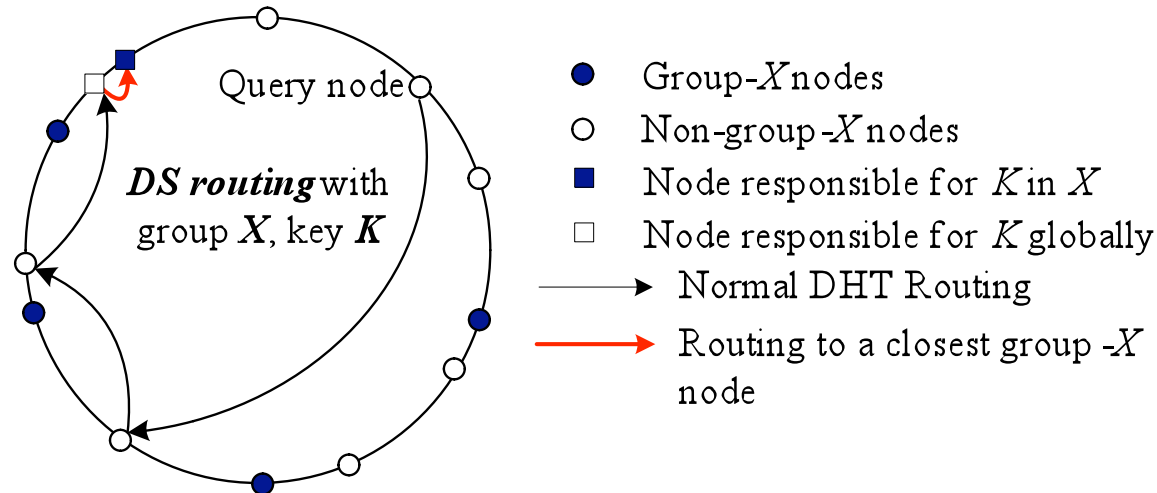
## ■ Group- $X$ PC routing with key $K$



- The path from the query node to the root for  $K$  in  $X$  is always constrained within  $X$
- PC routing algorithm is similar to Tapestry
  - but messages must be routed by group- $X$  links at each hop.

# Destination-Specified routing

- A possible implementation of group- $X$  DS routing

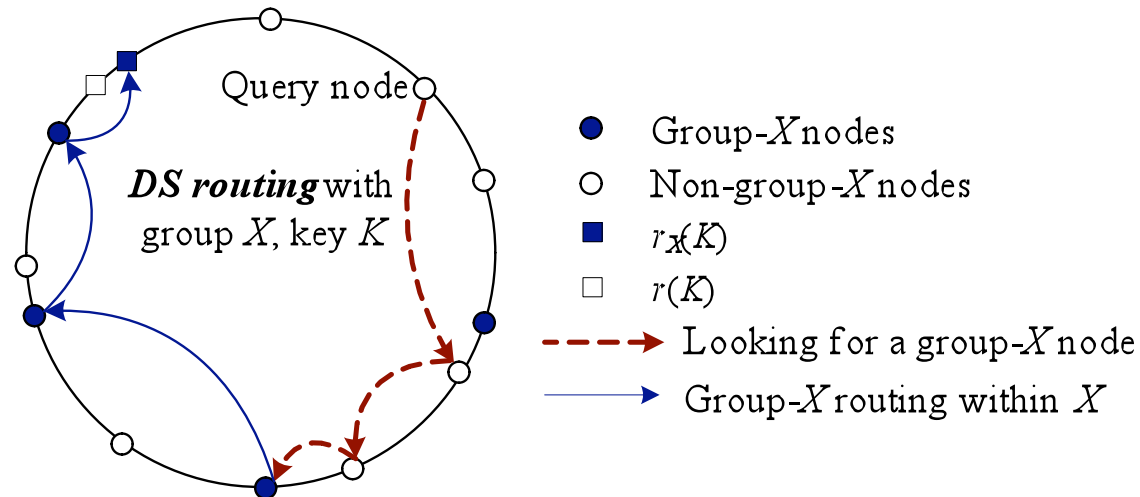


- route from the query node to the global root for  $K$ .
- route to the nearest group- $X$  node, which must be the group- $X$  root for  $K$
- We implement DS routing in another way (reuse PC routing)

# Destination-Specified routing

(cont.)

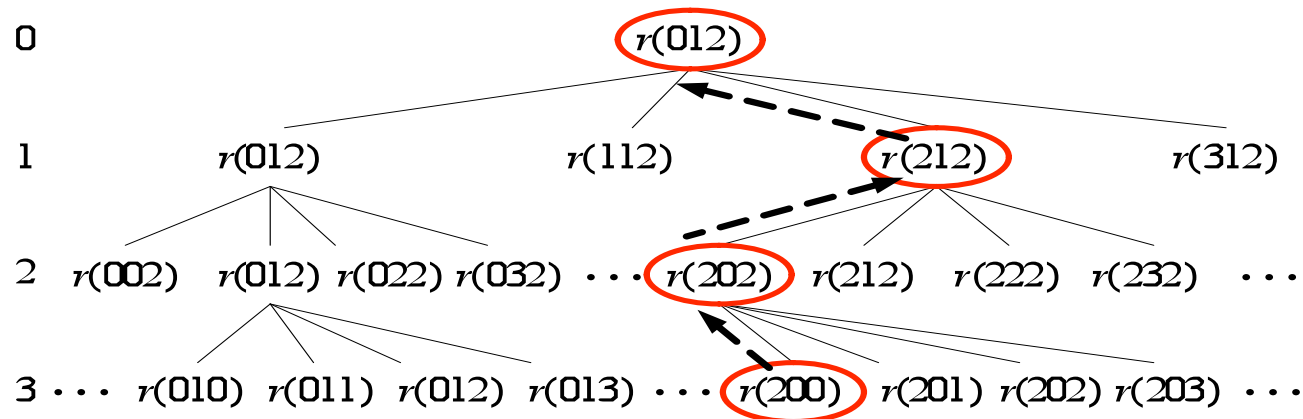
- G-Tap's implementation (reuse PC routing)



- Given a group  $X$  and a message with key  $K$ , the DS routing is conducted by
  - (i) Routing to an arbitrary group- $X$  node  $v$
  - (ii) Routing from node  $v$  to the group- $X$  root for  $K$ .

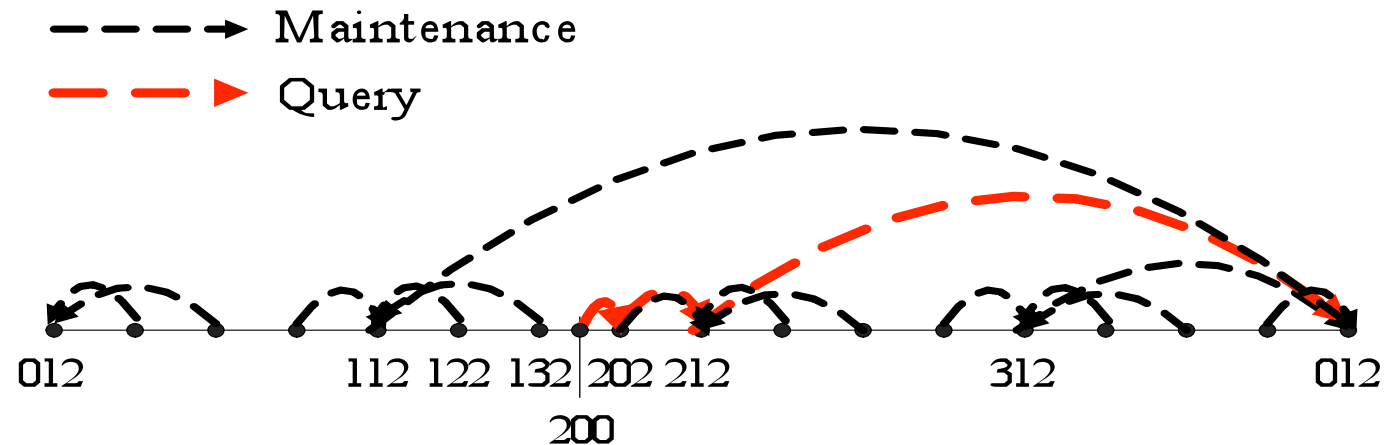
# Discover a group

- The way for looking for an arbitrary group- $X$  node
  - Walk along a bottom-up path from itself to the root of the GMR tree
  - Check the fingers of each node along the path, and
  - Stop once a group- $X$  finger is found
  - E.g. look for group  $Z$  from node 200 (Hash( $Z$ )=012)



# Discover a group (cont.)

- The benefit of the GMR tree-based solution



- The query for group  $Z$  from node 200 can be locally resolved whenever there is a group- $Z$  node nearby

---

# Discover a group (cont.)

- Above intuitional method has a problem
  - There are  $n+1$  levels in the GMR tree.
  - A query might travel at most  $n$  tree-hops before answered
- Solution
  - Note that a G-TAP node may exist at multiple levels in the tree, and it can always determine the highest level it exists
  - When a node receives a query message for group  $X$ 
    - directly forward the query to its parent at the highest level
  - GTap system with size  $N$ , base  $b$  and group size  $M$ 
    - Expected group discover latency is less than  $3\log_b(N/M)$

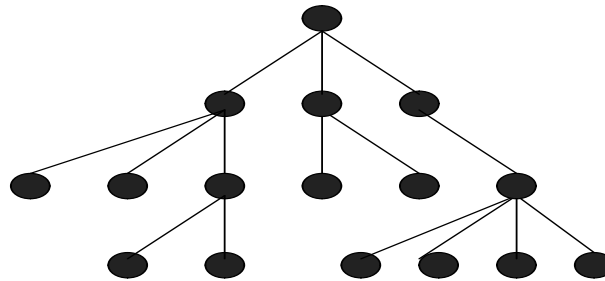
---

# Hierarchical DHTs (H-TAP)



# Hierarchical systems

- Hierarchy is common in large-scale systems
  - Hierarchical administrative domains
  - Universities, governments



- G-TAP
  - without concern for relationship between groups
- H-TAP
  - optimization of G-TAP for hierarchical groups
  - one group might be a subset of another

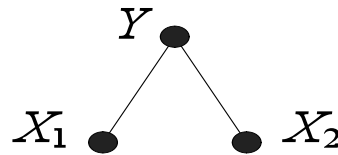
# Two-level hierarchy

- Simplest case of a two-level hierarchy

- Three groups  $X_1$ ,  $X_2$ , and  $Y$  satisfying

- $X_1 \subset Y, X_2 \subset Y,$

- $X_1 \cup X_2 = Y, X_1 \cap X_2 = \emptyset$



- Suppose there is a group- $Y$  node, say node  $u \in X_1 \subset Y$  without loss of generality

- Node  $u$  has entry set, leaf set and finger set

- The leaf set is the same as in G-TAP

- The finger set is determined by the two group- $X_1$  and group- $X_2$  GMR trees separately

---

# Two-level hierarchy (cont.)

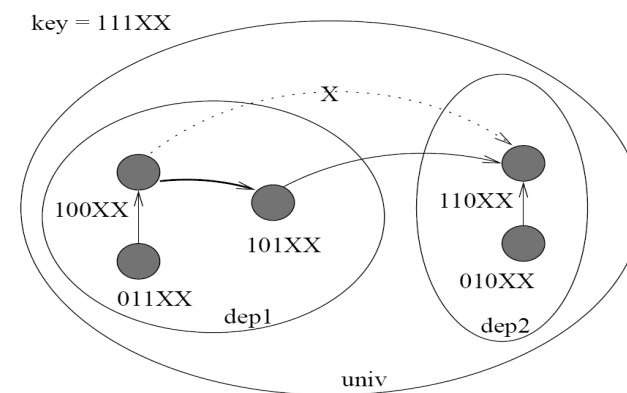
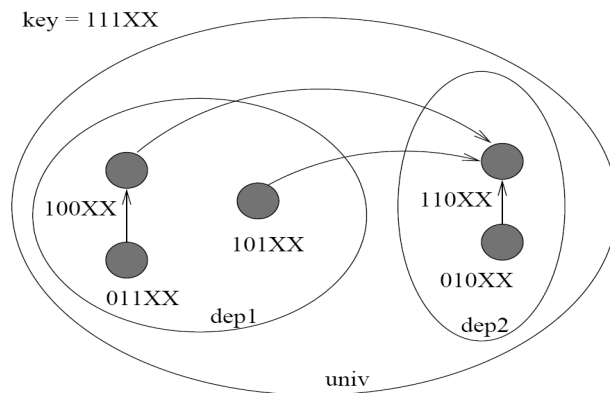
- Entry set of node  $u$ 
  - Contains group- $X_1$  links and group- $Y$  links
  - Group- $X_1$  links are the same as in G-TAP
  - Group- $Y$  links, i.e. group- $X_2$  links, are different
    - Besides the conditions in G-TAP, a group- $Y$  link exists in node  $u$ 's entry set only if it is closer to node  $u$  than any group- $X_1$  link
    - Similar to the design of a well-known domain-aware DHT: Canon [ICDCS04]

# Flexible routing

- PC/DS routing in the lowest level (group- $X_1/X_2$ )
  - Identical to that in G-TAP
  - Because there is no change in the lowest level routing structures
- Suppose a group- $Y$  PC routing for  $K$  from a group- $X_1$  node
  - Traverse group  $X_1$  until it arrives at the group- $X_1$  root for  $K$
  - Forwarded by a group- $Y$  (i.e. group- $X_2$ ) link.
  - Traverse group  $X_2$  until it reaches the group- $X_2$  root for  $K$ .
- Group- $Y$  DS routing for  $K$  can be conducted by
  - (i) routing to an arbitrary group- $X_1/X_2$  node  $v$
  - (ii) routing from  $v$  to the group- $Y$  root for  $K$ .

# Path locality/convergence

- H-TAP supports path locality/convergence.
  - Locality: The route between two nodes never leaves the lowest-level domain that contains both nodes.
  - Convergence: The routes from different nodes in a domain  $D$  to a node  $x$  outside  $D$  converge at a common node  $y$  in  $D$  and then exit  $D$ .



---

# Path locality/convergence (cont.)

- How H-TAP supports path locality/convergence
  - A message with key  $K$  originates from a node  $A$ , user1.research.intel.com; and the root for  $K$  in intel.com is another node  $B$ , user2.market.intel.com
  - The PC routing within intel.com guarantees
    - Locality: the route cannot leave intel.com until it reaches  $B$  (and thus will never pass through nodes like guy.amd.com)
    - Convergence: the roots for  $K$  in research.intel.com and intel.com are just the convergence points for the two domains

---

# Conclusion

- G-TAP/H-TAP support to organize nodes into groups and to control DHT routing path and destination
- Questions?



---

# BACKUPS



---

# Backups

- Why not just use many DHTs, one for a group?
  - One DHT is more convenient for upper-layer applications
  - Tapestry has  $c$  links at each entry, if we have  $b$  groups ( $b \leq c$ ),  $b$  Tapestry will have  $c*b$  links at each entry, while G-TAP has  $c$  links

# Backups

- What's the difference of G-TAP/H-TAP from Diminished Chord or Domain-aware DHTs?

	G-TAP/H-TAP	Domain-aware DHTs	D-Chord
Group formation	Yes		Yes
Hierarchical structure	Yes	Yes	
Traditional routing	Yes	Yes	Yes
DS routing	Yes		Yes
PC routing	Yes		
Path locality	Yes	Yes	
Path convergence	Yes	Yes	

---

# Backups

- What if the number of groups to which a node belongs is very high (e.g. 1000)?
  - Definitely the routing table size would be very large
  - It is an inevitable tradeoff
  - We will study this problem deeply in the future.

---

# PROPERTIES



---

# Configuration

## ■ G-TAP

- ❑ 64/256 groups, a node joins in 0~4 groups
- ❑ The base DHT is also viewed as a group, the average number of groups to which a node belongs is 3

## ■ H-TAP

- ❑ The number of levels in the hierarchical tree is 3
- ❑ The branching factor ( $bf$ ) of internal nodes is 8/16
- ❑ Same number of lowest level of groups (units), and same average number of groups of a node

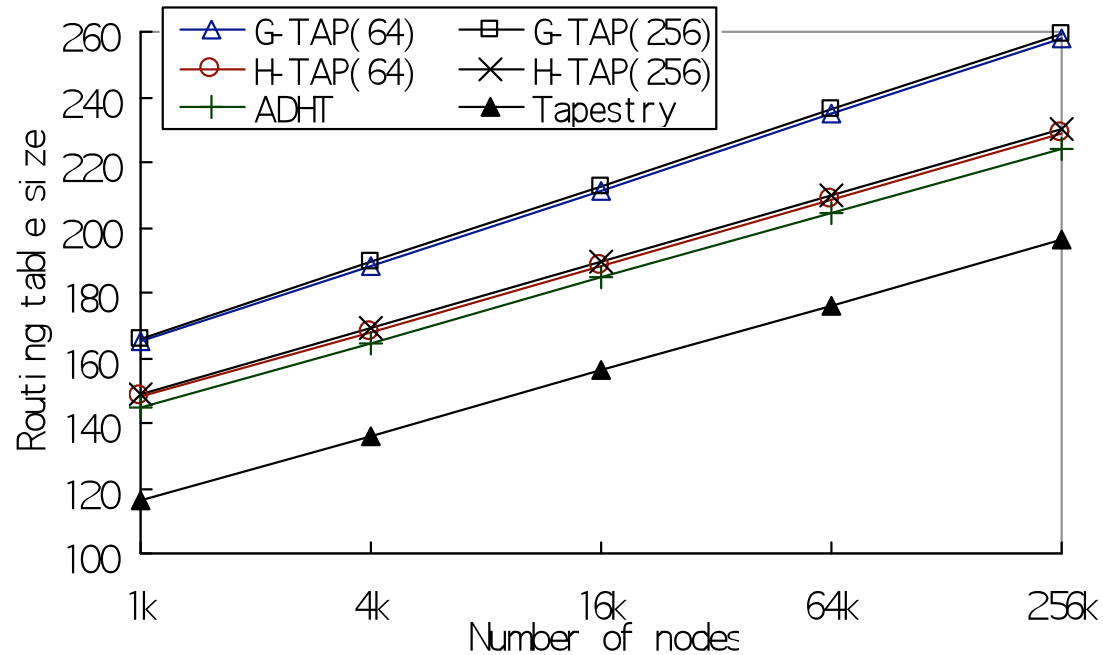
---

# Configuration (cont.)

- ADHT (a variation of Pastry similar to Tapestry)
  - Entry set is similar to that in Pastry/Tapestry
  - $2f$  leaves for each level of domain it belongs to
  - Uses DHT *put/get* interfaces for domain discovery
  - Same configuration for H-TAP
- Common parameters
  - backup factor  $c = 3$ ,  $|L| = 2f = 16$ .
  - The number of nodes is varied from 1K to 256K.
  - Random nodeID of base  $b = 16$  and length  $n = 8$

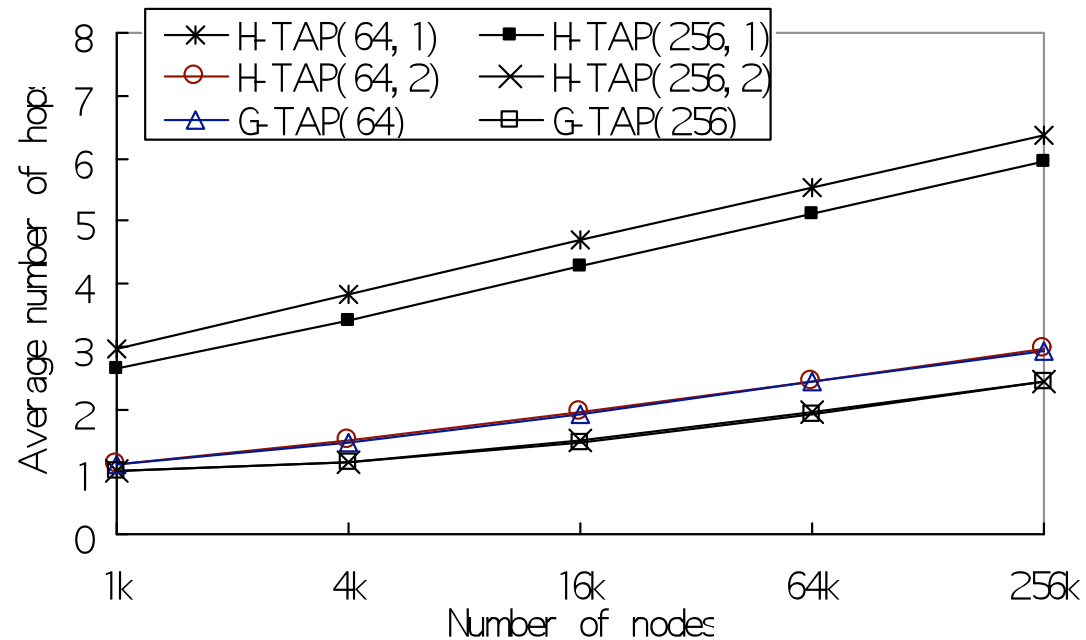
# Routing table size

- Including the entry set, leaf set, and finger set



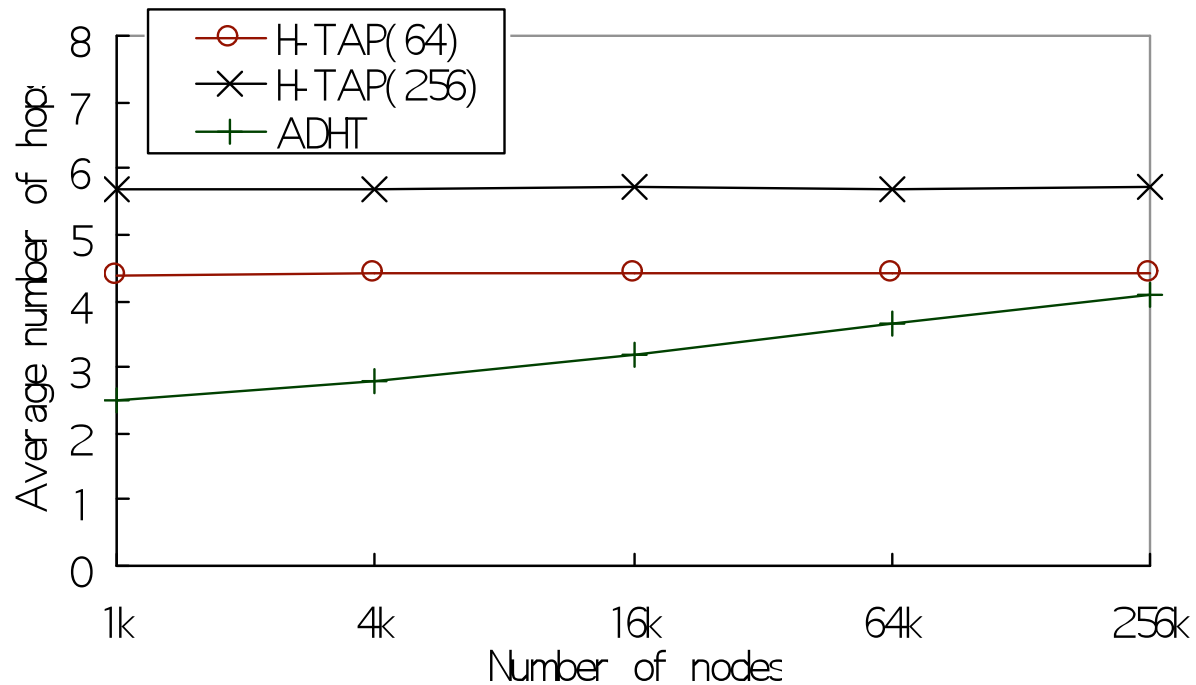
# PC routing

- PC routing in G-TAP and H-TAP
  - For H-TAP we evaluate the PC routing both within level-1 groups and within level-2 groups (i.e. units)



# Group discovery

- Group discovery in H-TAP and ADHT (put/get)



# Attaining Path locality/convergence

- H-TAP V.S. ADHT (support locality/convergence)

