DGRID: A DHT-Based Resource Indexing and Discovery Scheme for Computational Grids

Verdi MARCH, **Yong Meng TEO**, and Xianbing WANG

Department of Computer Science
National University of Singapore
Email: teoym@comp.nus.edu.sg

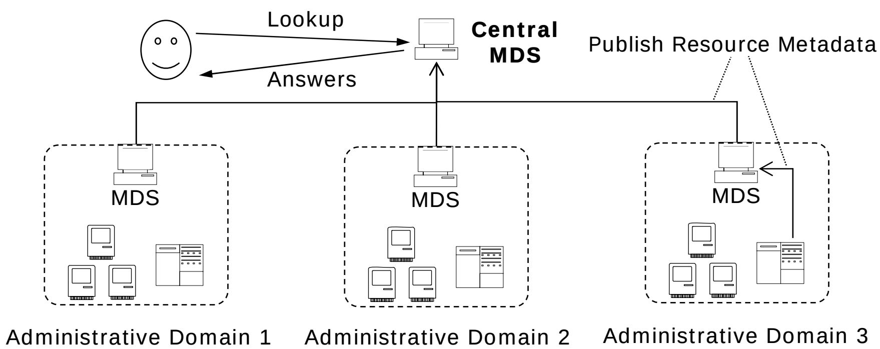
Outline

- **Introduction**
- DGRID
- Performance Evaluation
- Related Work
- Conclusion

Introduction

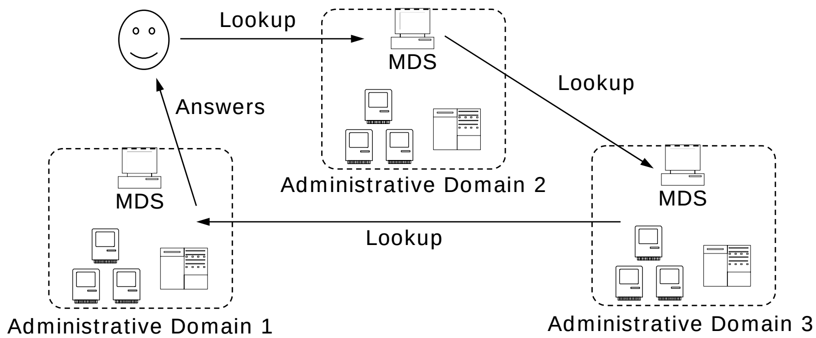
- Computational grid facilitates sharing of compute resources across administrative domains
- Resource discovery is an important infrastructure
  - Grid user searches for specific resources
  - Grid user deploys application on resources discovered earlier

Centralized Resource Discovery



Central MDS is a potential bottleneck and single point of failure

Distributed Resource Discovery



- Reduce dependency to 3rd party central MDS
- Administrative domains form an overlay network

DHT-based resource discovery

P2P Lookup

Property	Centralized	Decentralized	
		Unstructured	DHT
Data-item distribution	Yes	Optional	Yes (+mapping)
Single point of failure	Yes	Minimized	Minimized
Result guarantee	High	High for popular data items only	High
Lookup path length	One hop	$\Theta(N)$	$O(\log N)$ hops
Scalability of Lookup	Significant investment	Trade-off with result guarantee	Yes
Overhead of maintaining overlay	None	Lower	Higher

Distributed Hash Table

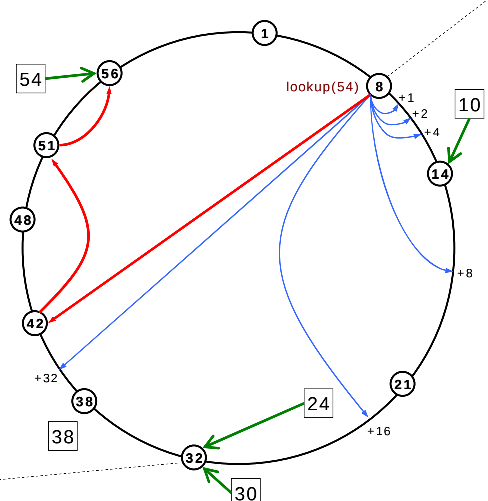
- DHT aims to achieve:
  - lookup with high result guarantee
  - short bounded lookup path length
  - scalability as number of nodes increases
- Hash-table-like interfaces
  - Key-value pair: `<SHA1("a.mp3"), "http://peer.id/a.mp3">`
  - ***join*** operation : *join*(*node*)
  - ***store*** operation : *put*(*key*, *value*)
  - ***lookup*** operation : *get(key)*

Distributed Hash Table (DHT)

m = 6-bit

- k = 0 ... 63 (square)
- n = 0 ... 63 (circle)

Chord is a DHT implementation from MIT



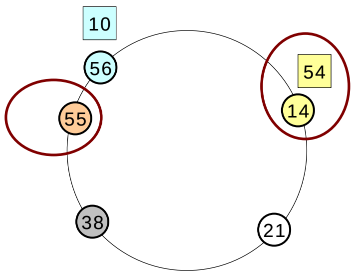
Finger table

8+1	14
8+2	14
8+4	14
8+8	21
8+16	32
8+32	42

Key-value pairs

Key	Value	Owner
24	1.mp3	8
30	a.pdf	21
30	a.pdf	48

Data-Item Distribution



Placement of data items are determined by key-to-node mapping function, i.e. no ***autonomy*** in placing data items

Key Issues:

- **Ownership**
  - data items are valuable assets (e.g. indexes of a search engine)
- **Self-interest of nodes**
  - protect investment of storage, bandwidth
  - Admin. domain may not want to index resources belonging to other admin. domains (e.g. would it profitable to "help" competitor by processing their queries?)
- **Write access** introduces security risk
- **Stale data items**, lookup resiliency

Outline

- **Introduction**
- **DGRID**
- Performance Evaluation
- Related Work
- Conclusion

Design Considerations

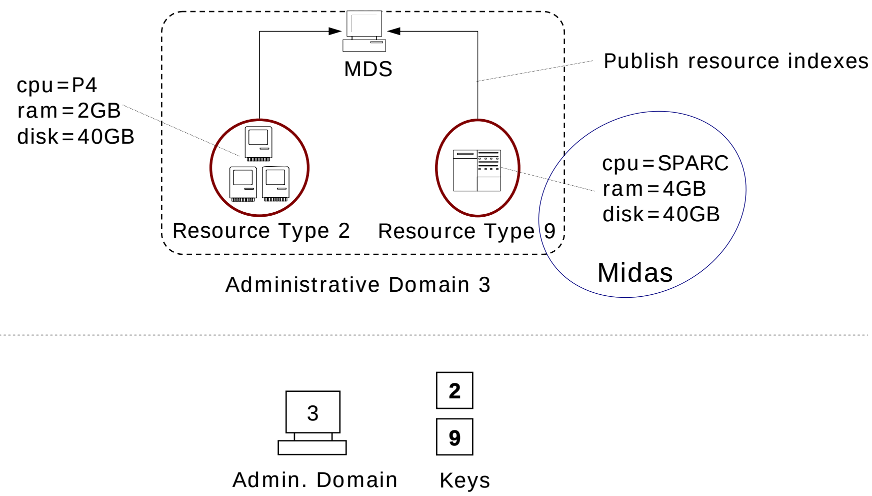
- Administrative domains store its own resource indexes
- Each *resource type* consists of resource instances with same attributes (e.g. CPU, RAM, etc.)
- Each resource type can be shared by many administrative domains
- # resource types are smaller than in P2P file sharing

Design

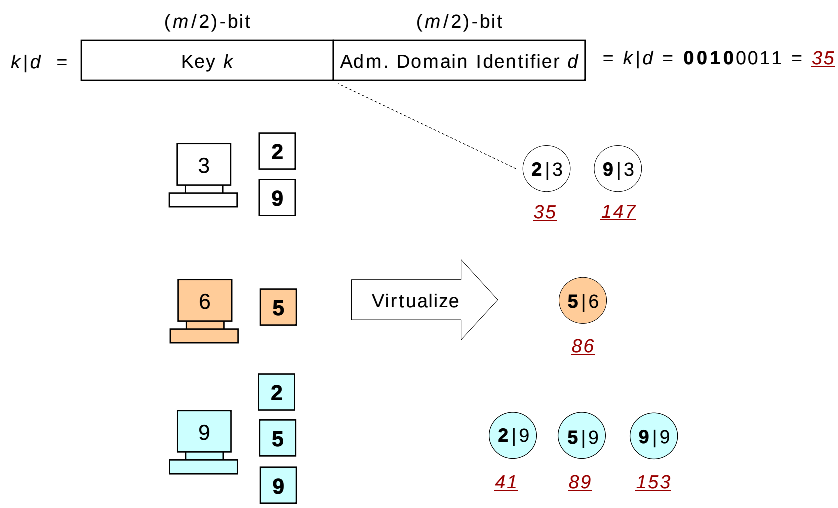
- Based on *R*ead-only *D*istributed *H*ash *T*able (R-DHT)
  - Keys are mapped to their original location
  - Each node provides read-only access (because it stores only its own key-value pairs)

Hash-Table Interface	Conventional DHT	R-DHT
put(*key*, *value*)	Yes	No
get(*key*)	Yes	Yes
join(*node*)	Yes	Yes

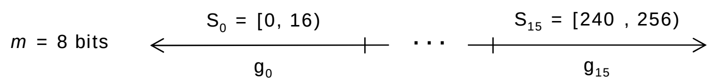
R-DHT Terminologies

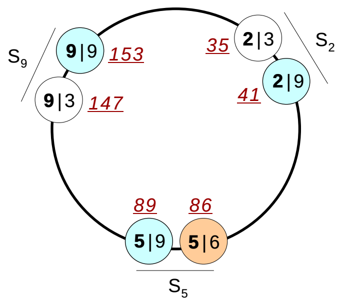


Read-Only Mapping



R-Chord





N admin. domains
T unique keys per host (mean)
K unique keys

NT nodes
K segments

Lookup path length = O(log *NT*)

- Longer than Chord

Lookup Optimizations

- Lookup path length = **O(min(log *K*, log *N*))**
  - *Routing by segments*
    - Locating a key (resource type) = locating a segment
    - O(log *K*) hops
  - *Shared finger tables*
    - O(log *N*) hops even if *K* > *N*
    - Rationale: visit 1 host = visit T_d nodes
- Many admin. domains share the same resource type
  - Increase lookup resiliency without introducing stale data

Outline

- Introduction
- DGRID
- **Performance Evaluation**
- Related Work
- Conclusion

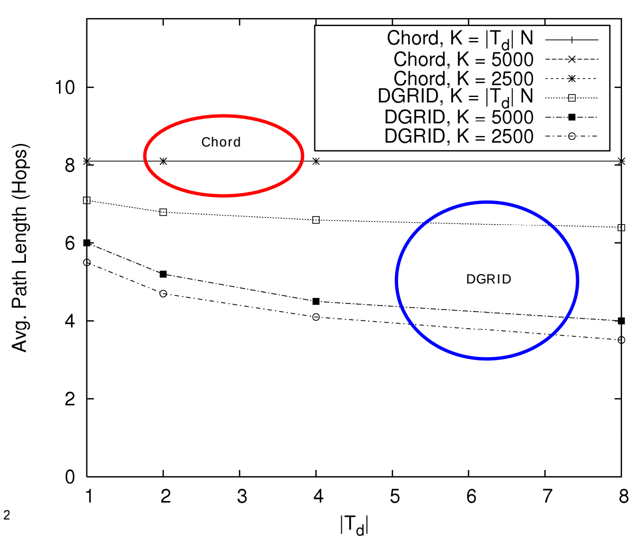
Theoretical Analysis

Property	Chord	R-Chord
Lookup path length	$O(\log N)$	$O(\min(\log K, \log N))$
Join	$O(\log_2 N + \lvert T_d \rvert \log N + K \ln K/(K-T))$	$O(\lvert T_h \rvert \log^2 V)$
#fingers per host	$O(\log N)$	$O(\lvert T_h \rvert \log V)$
Finger flexibility	$O(1)$	$O(N \ln K/(K-T))$
Stabilization	$O(N \log^2 N)$	$O(V \log V \log N)$
Add new key	$O(\log N)$	$O(1)$
Add data item with existing key	$O(\log N)$	$O(\log^2 V)$
Update data item	$O(\log N)$	$O(1)$
#data items (incl. replication)	$O(N \ln K/(K-T) \log N)$	$O(N \ln K/(K-T))$

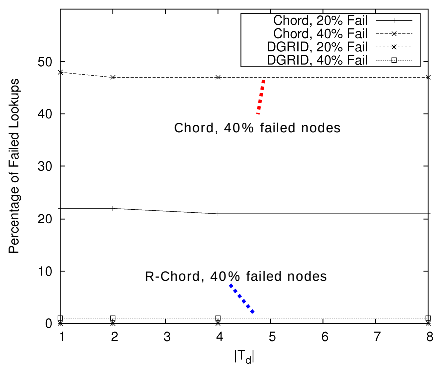
Simulation Analysis

- Based on Chord simulator
- Parameters:
  - *N* = # admin. domains
  - *K* = # resource types
  - *T* = average # resource type per admin. domain
  - $|T_d|$ = # resource type shared by admin. domain *h*
- Admin. domain inter-arrival rate is Poisson inter-arrival rate

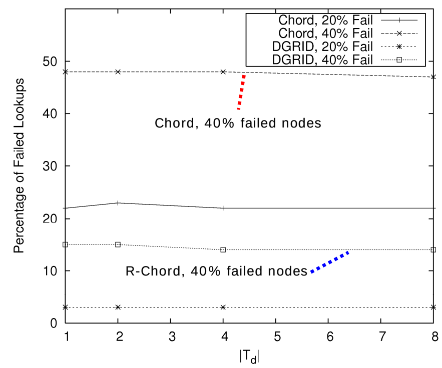
Lookup Path Length



Resiliency to Simultaneous Node Failures



K < N



K > N

Outline

- Introduction
- DGRID
- Performance Evaluation
- **Related Work**
- Conclusion

Related Work

- Distributed grid resource discovery
  - Based on unstructured overlay network: Routing-transferring model (Li et. al., 2002) and Iamnitchi 2002
  - Based on DHT: self-organizing Condor pools (Butt et. al., 2003), XenoSearch (Spence et. al., 2003), MAAN (Cai et. al., 2004), RIC (Zhu et. al., 2004)
- DHT
  - R-DHT generalizes SkipGraph (Aspnes et. al., 2003)
  - SkipNet (Harvey et. al., 2003) supports more flexible data placement, but this is not exposed using hash-table lookup interface
    - "retrieve `index.html` from any node under domain `nus.edu.sg`."

Outline

- Introduction
- DGRID
- Performance Evaluation
- Related Work
- **Conclusion**

Conclusion

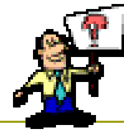
- DGRID is a DHT-based resource indexing and discovery where each administrative domain stores its own resource indexes
- Lookup path length is shorter than conventional DHT
- Increased lookup resiliency by exploiting the property that many administrative domains share the same resource types

Ongoing Work

- Address overhead of maintaining overlay network
  - Hierarchical DGRID
  - Dynamic per-node routing table
- Multi-attribute resource indexing and based on space-filling curve (SFC)
- Selective replication
- ….

Papers

www.comp.nus.edu.sg/~teoym



1. V. March, **A Read-only Distributed Hash Table**, PhD Thesis, Department of Computer Science, National University of Singapore, December 2006 (under examination).

2. V. March and Y.M. Teo, **Multi-Attribute Range Queries on Read-only DHT**, Proc. of 15th International Conference on Computer Communications and Networks, pp. 419-424, IEEE Communications Society Press, USA, October 2006.

3. G. Ghinita and Y.M. Teo, **An Adaptive Stabilization Framework for Distributed Hash Tables,** Proc. of 20th International Parallel & Distributed Processing Symposium, IEEE Computer Society Press, Greece, April, 2006.

4. V. March, Y.M. Teo, H.B. Lim, P. Eriksson and R. Ayani, **Collision Detection and Resolution in Hierarchical Peer-to-Peer Systems,** Proc. of 30th IEEE Conference on Local Computer Networks, IEEE Computer Society Press, pp. 2-9 , Sydney, Australia, November 2005.