

Hierarchical P2P Systems

Luis Garcés-Erice

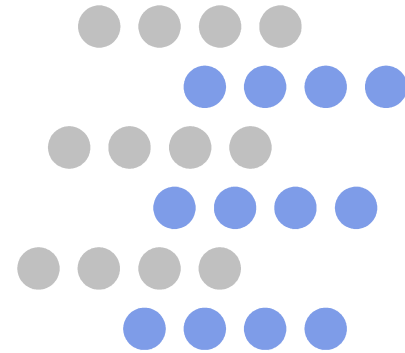
Ernst W. Biersack

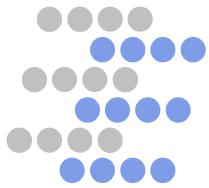
Pascal A. Felber

Keith W. Ross

Guillaume Urvoy-Keller

Institut Eurécom – Sophia Antipolis, France





Outline

Introduction to DHTs and Hierarchical P2P

Chord

Hierarchical Framework

Hierarchical DHTs

Hierarchical Lookup

Hierarchical DHT Management

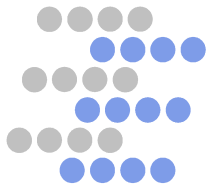
Advantages of Hierarchical DHTs

Caching Infrastructures

Transparency and Autonomy

Efficient Lookup

Conclusion



Introduction (I)

P2P Lookup Services take care of assigning and locating resources in P2P systems.

Structured P2P systems:

A “key” is a resource identifier.

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

$h(\text{key}) \rightarrow 0, 1, 2 \dots N-1$, all N possible outputs of $h()$

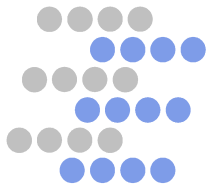
In Distributed Hash Tables (DHTs) each bucket is a peer.

A key is assigned to the peer with the “closest” id to the key.

Chord, CAN, Pastry are all flat DHTs, peers are all equal.

To be scalable, each peer knows a subset of all peers

$h(\text{key})$ must be **routed** to corresponding “bucket”



Introduction (II)

Chord DHT:

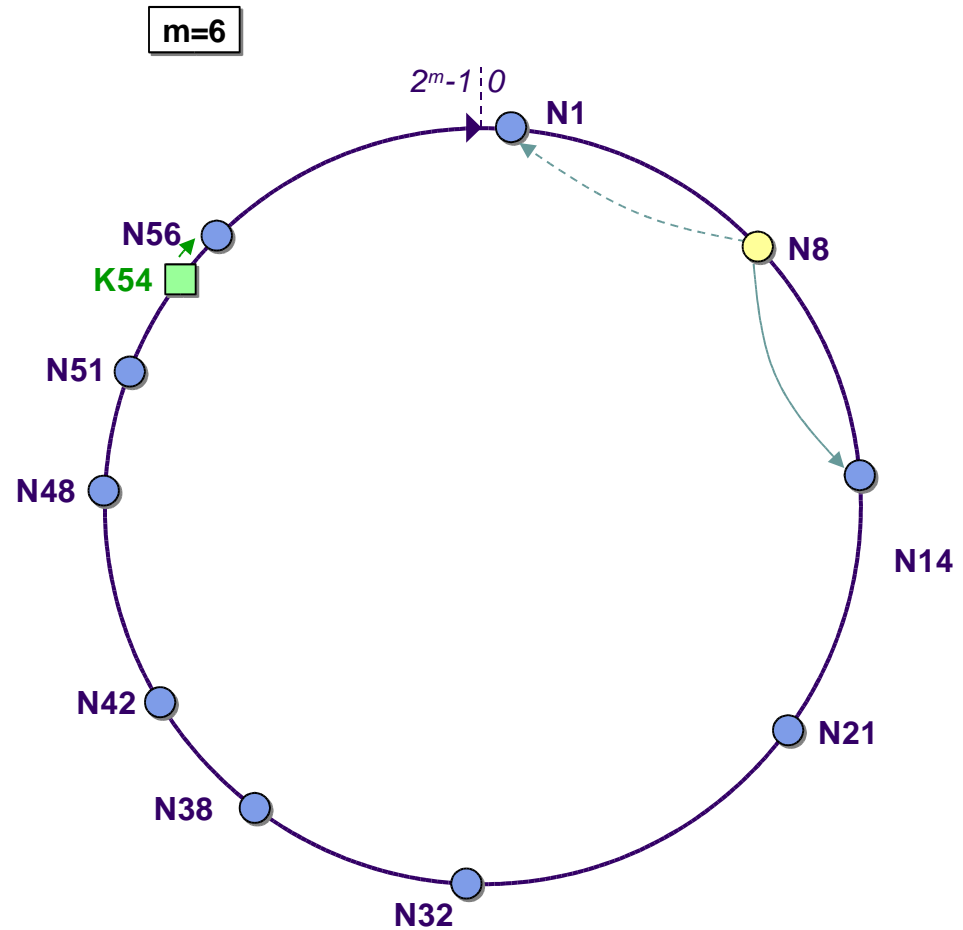
Each peer (and key) has a **m-bit id**

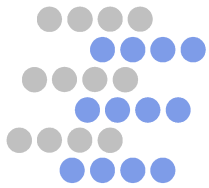
Peers are ordered by id in a modulo 2^m circle

A peer **p** is the **successor** of an id if **p** is the first existing peer on the circle **after** the id

A peer **p** is the **predecessor** of an id if **p** is the first existing peer on the circle **before** the id

O(N) hops routing





Introduction (III)

Chord DHT:

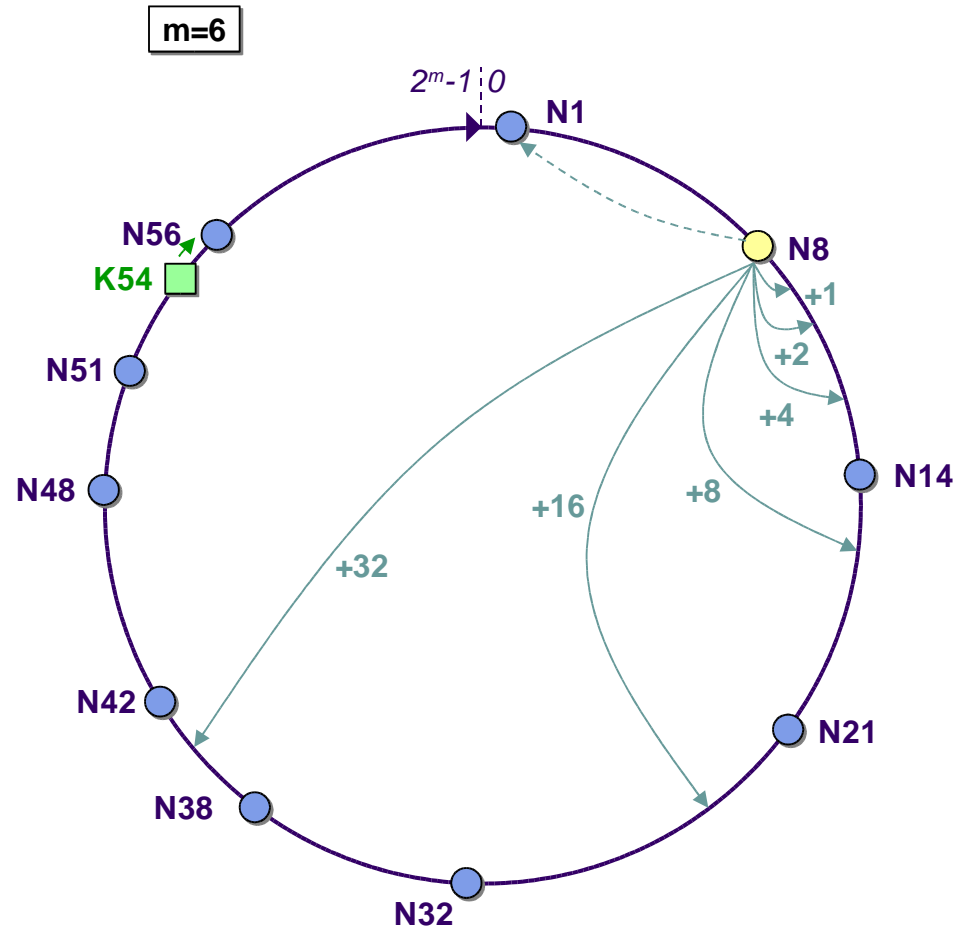
Each peer p knows a set of other peers
(**routing table**):

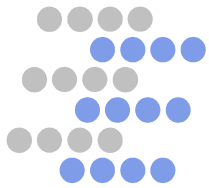
p 's successor

For $j=1, \dots, m$, the
successor of $p+2^{j-1}$,
called **fingers**

p 's predecessor

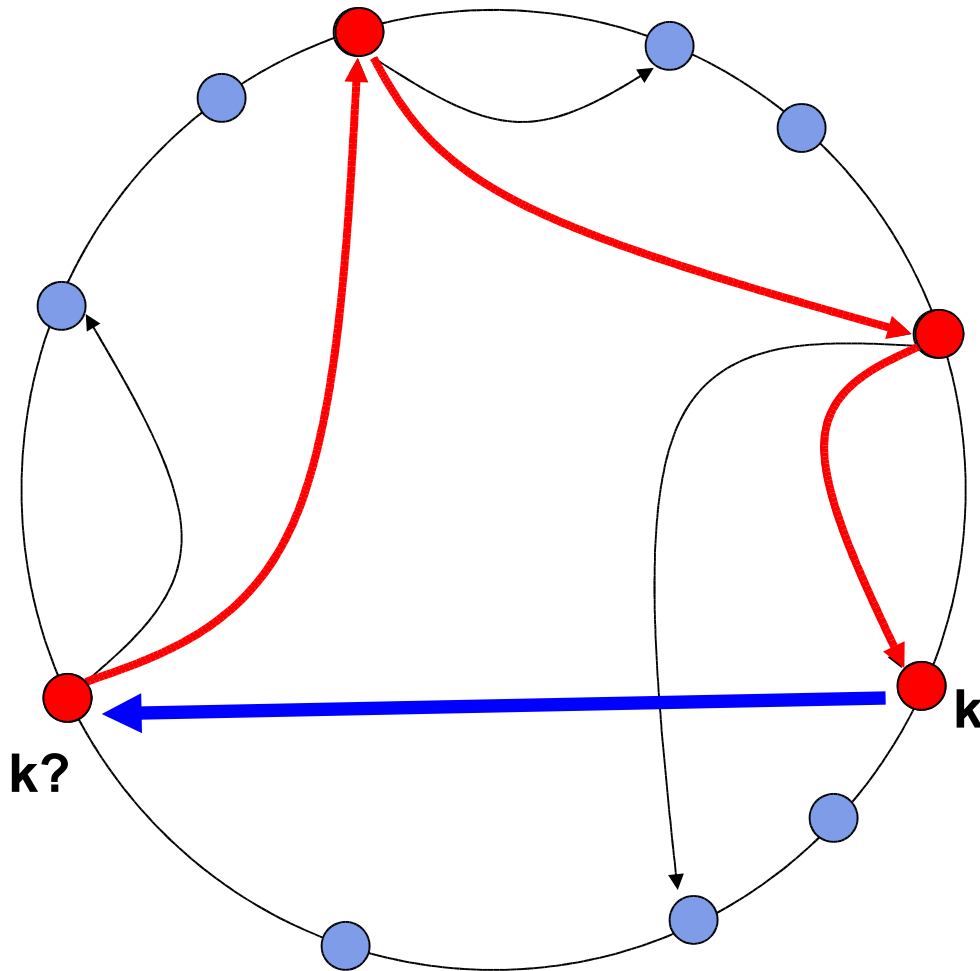
$O(\log(N))$ hops
routing



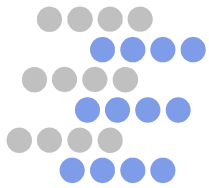


Introduction (IV)

DHT P2P key Lookup



1. Lookup of key k
1. Each peer **forwards** the query to the peer in its routing table **closest** to the key
1. Found **responsible** peer, transfer of resource corresponding to key



Introduction (V)

Very different from the Internet, where we find hierarchical routing:

- BGP to route from AS to AS

- Autonomous intra-AS routing (OSPF, RIP, ...)

- Scalable architecture, administrative autonomy

Number of hops needed to look up a key measures the efficiency of a Lookup Service

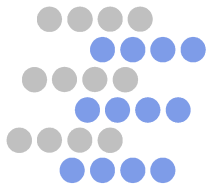
$O(\log(N))$ hops: What happens when peers have heterogeneous capabilities ? Can a hierarchical structure help ?

We study two-tier DHTs:

- Peers are organized in disjoint groups

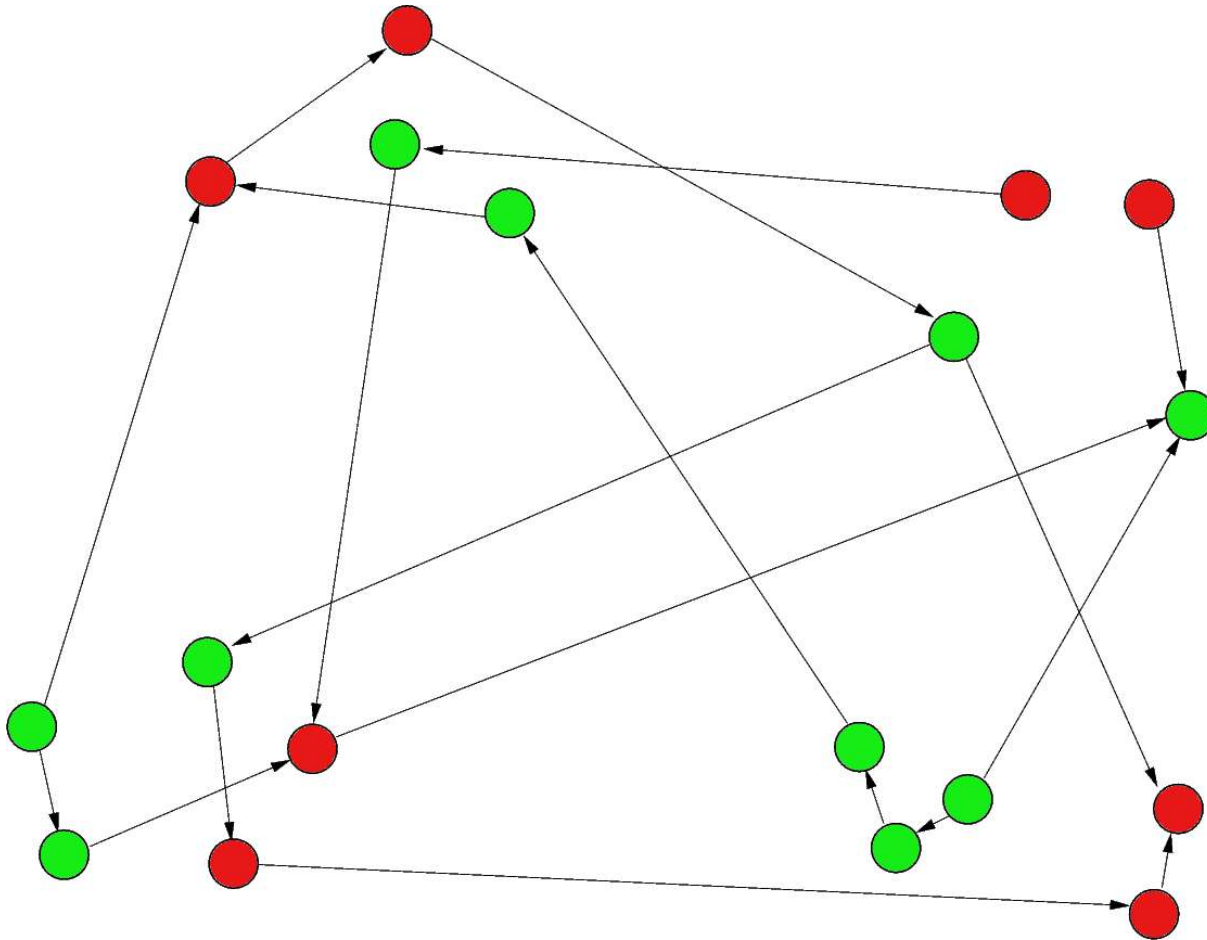
- Lookup messages are first routed to a group, then to a peer in that group

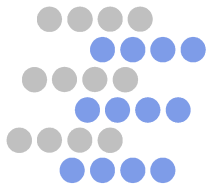
Extension to a K-tier DHT is straightforward



Hierarchical Framework (I)

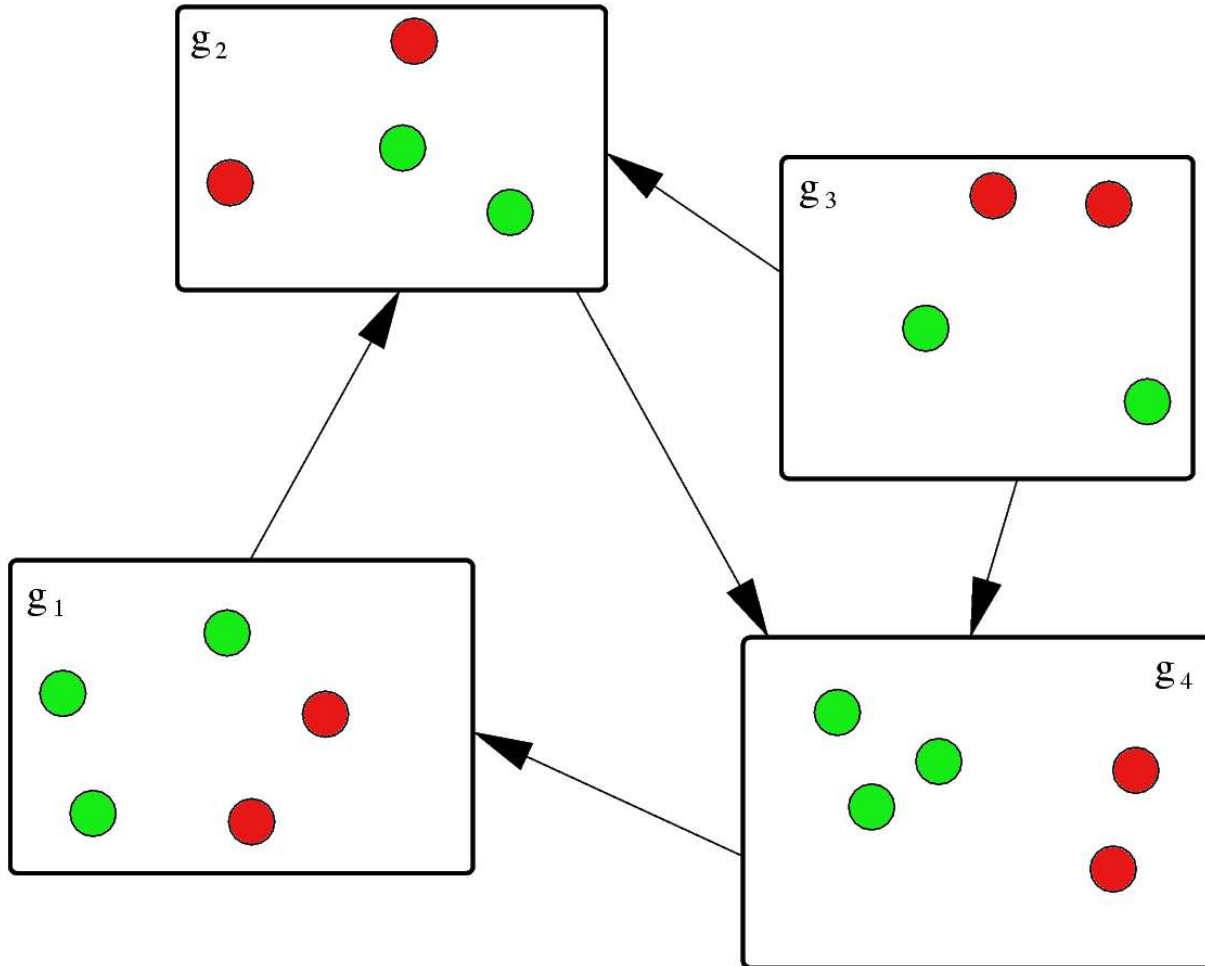
Normally, DHTs are flat structures of heterogeneous peers

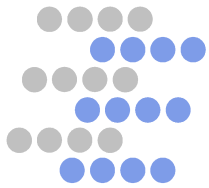




Hierarchical Framework (II)

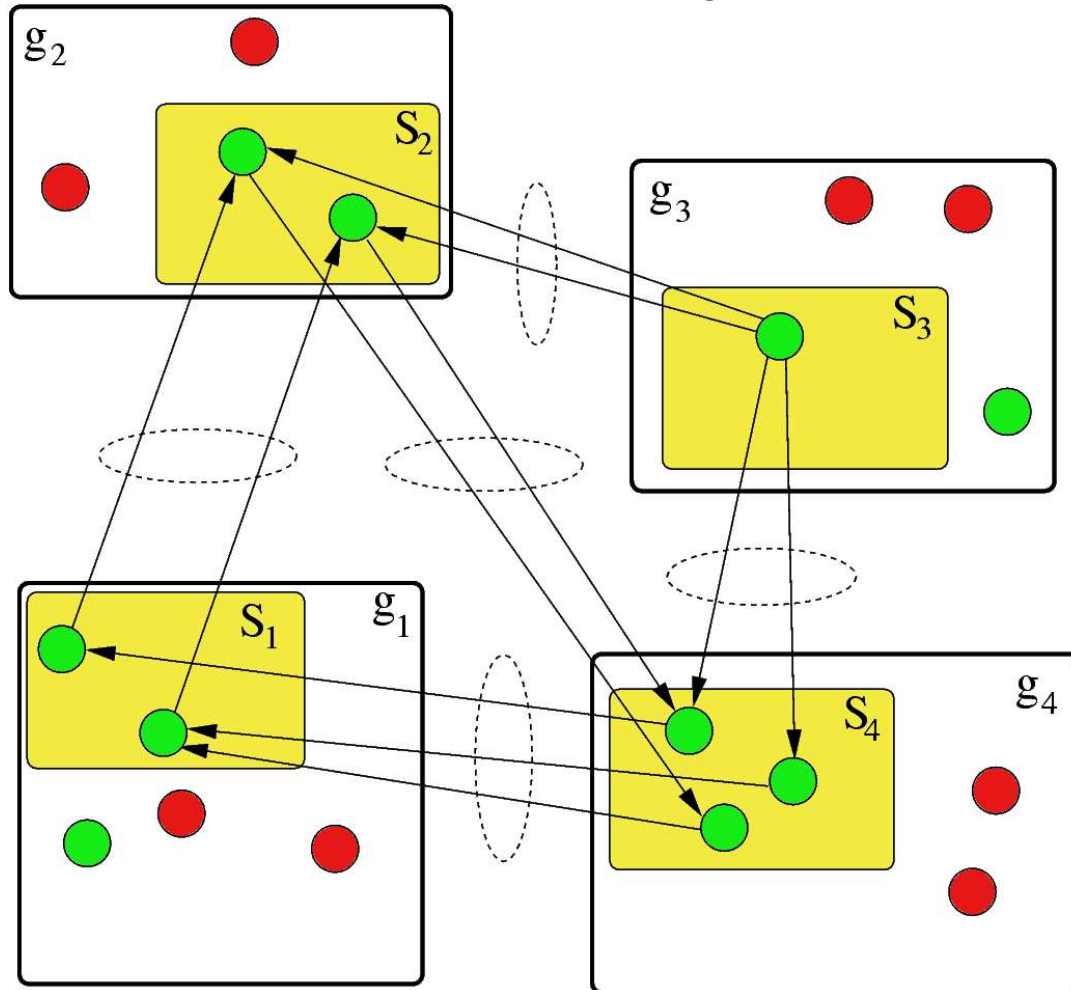
In Hierarchical DHTs, peers are organized in groups

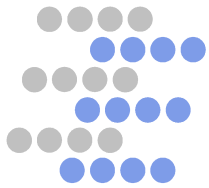




Hierarchical Framework (III)

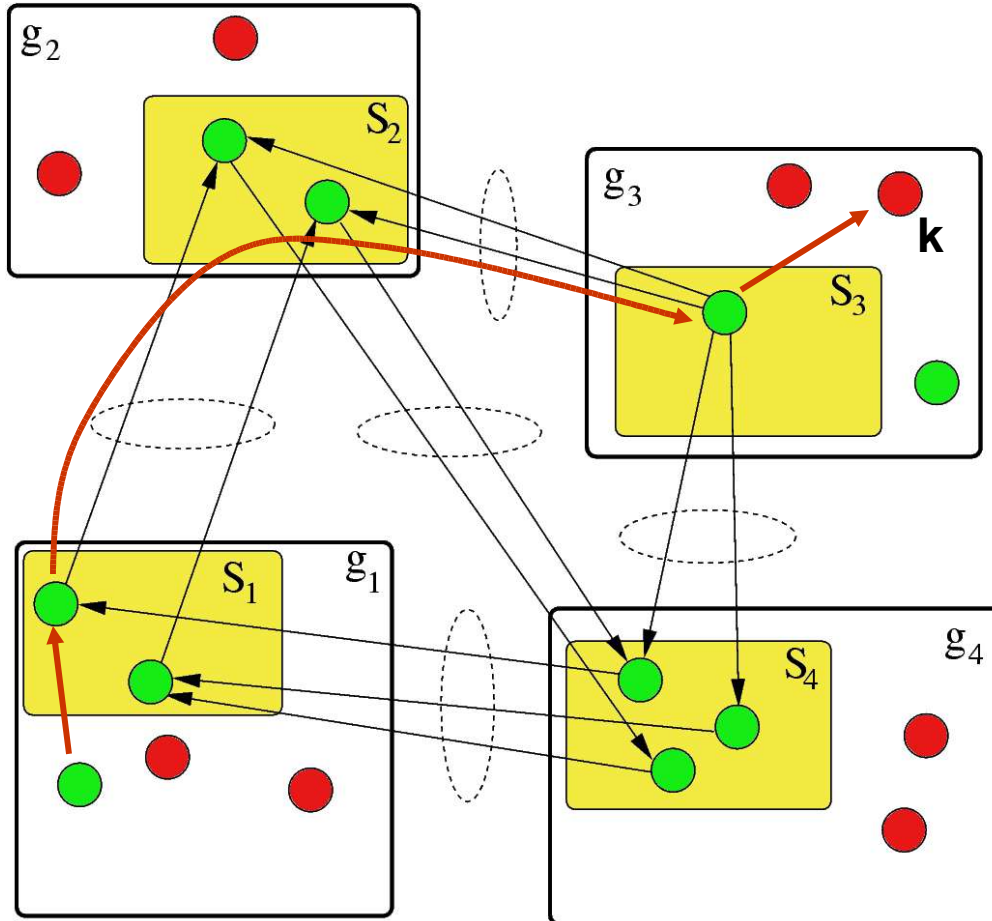
Peers can have different roles in groups



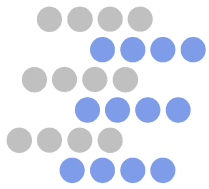


Hierarchical Lookup

Group g_3 responsible for key k



1. Find responsible **group**
1. Find responsible **peer** in group



Hierarchical DHT Management (I)

In groups with c superpeers, the first c peers to join will be the superpeers

When a peer joins a group, it first contacts one superpeer

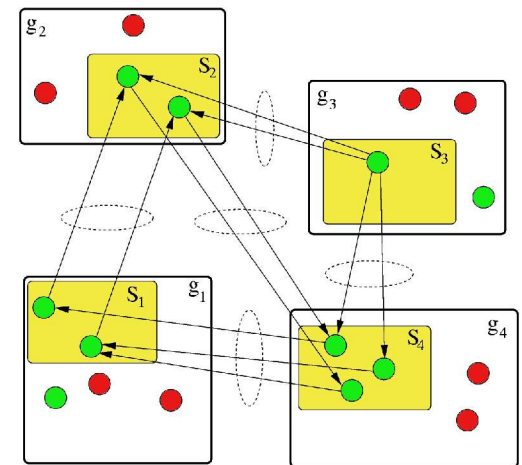
Peers must present their characteristics to the superpeer

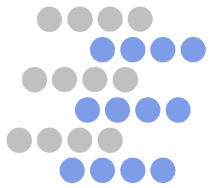
Best peers must be monitored by superpeers

Those best peers with highest up-time (most stable) are promoted to superpeer

Failed superpeer is rapidly substituted

We provide stability to top-level overlay





Hierarchical DHT Management (II)

Each peer p_i entering the network must know:

Another peer p_f already in the network

Its own id p_i

Its group's id g_i

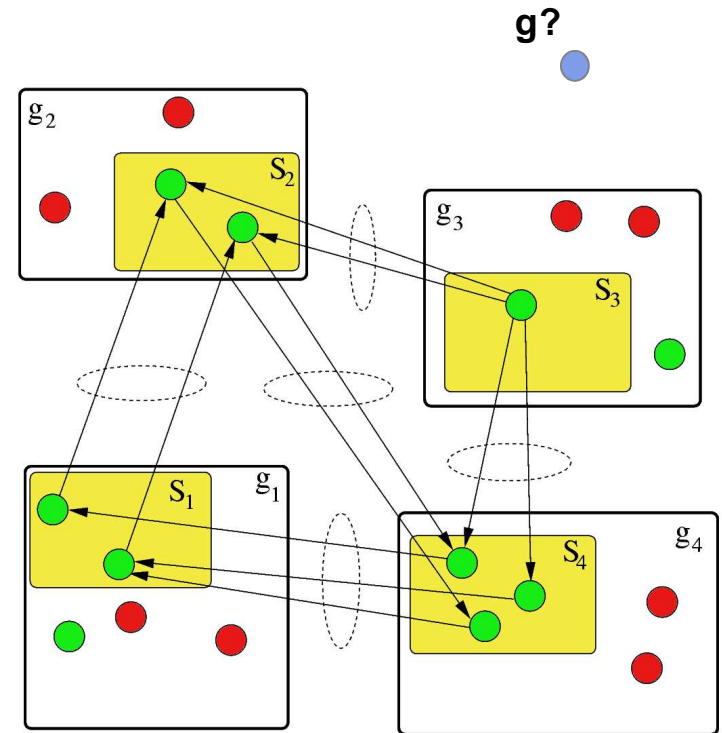
Peer p_i joins the DHT:

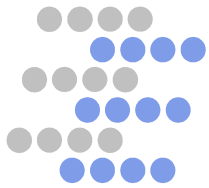
p_i asks p_f to lookup key g_i to find group g_i

If the group does not exist, insert p_i as new group g_i

Else, use the contacted peer in g_i to join group's overlay

Recursively for K-tier systems





Advantages of Hierarchical P2P

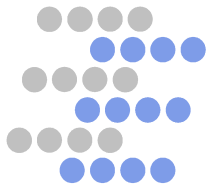
Exploits heterogeneous characteristics of peers

Less traffic in the wide-area

Natural caching infrastructure

Transparency and autonomy of different parts of the system

More efficient lookup of keys



Advantages of Hierarchical P2P

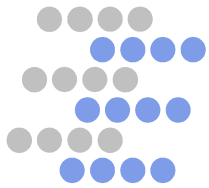
Exploits heterogeneous characteristics of peers

Less traffic in the wide-area

Natural caching infrastructure

Transparency and autonomy of different parts of the system

More efficient lookup of keys



Advantages of Hierarchical P2P

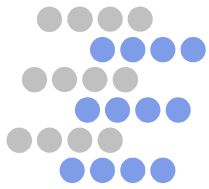
Exploits heterogeneous characteristics of peers

Less traffic in the wide-area

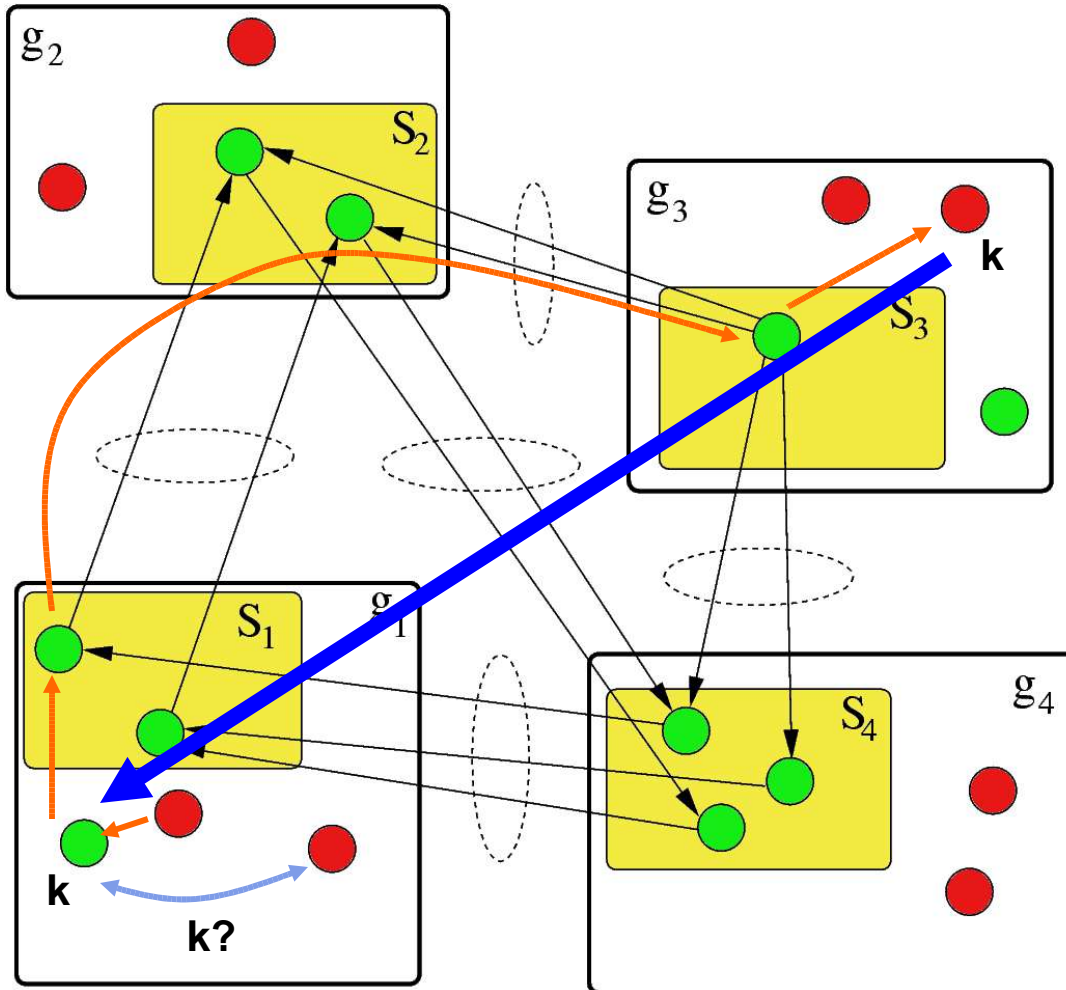
Natural caching infrastructure

Transparency and autonomy of different parts of the system

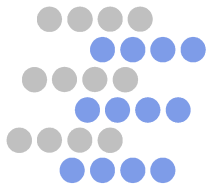
More efficient lookup of keys



Caching Infrastructure



1. Find responsible peer for k in its **own** overlay
1. This **last** looks up the key and **retrieves** the resource
1. Subsequent request is satisfied **inside own** group



Advantages of Hierarchical P2P

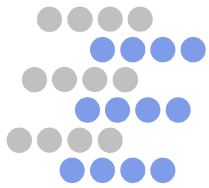
Exploits heterogeneous characteristics of peers

Less traffic in the wide-area

Natural caching infrastructure

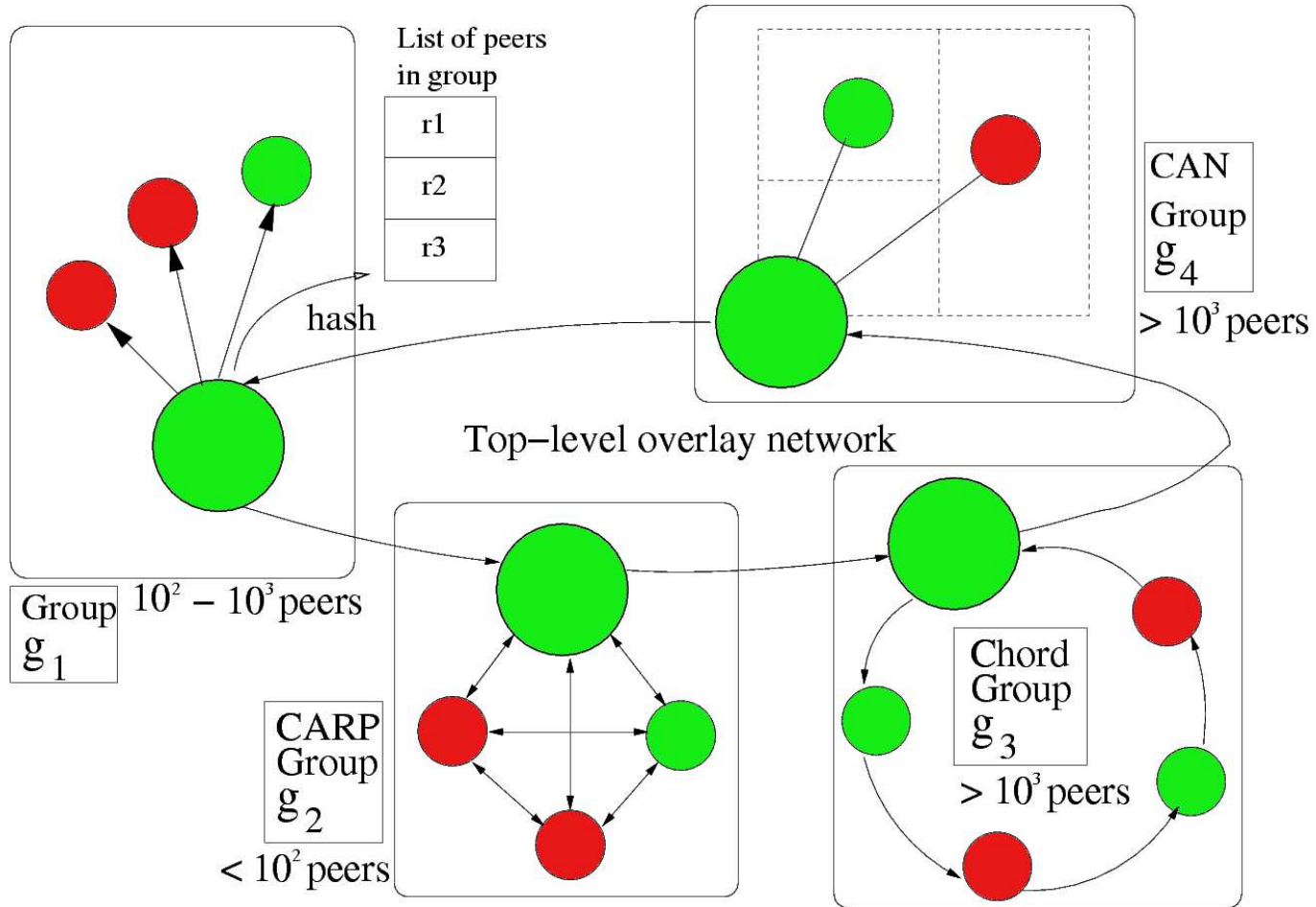
Transparency and autonomy of different parts of the system

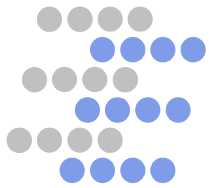
More efficient lookup of keys



Transparency and Autonomy

Groups can be managed independently





Advantages of Hierarchical P2P

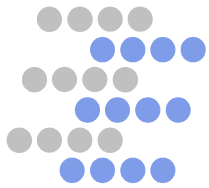
Exploits heterogeneous characteristics of peers

Less traffic in the wide-area

Natural caching infrastructure

Transparency and autonomy of different parts of the system

More efficient lookup of keys



Efficient Key Lookup (I)

We compare the average number of hops needed to find a key

- in a Chord ring

- in a hierarchical DHT with Chord in the top-level overlay

Two kinds of peers:

- Stable: peer is down with probability p_s . Few peers are stable

- Unstable: peer is down with probability p_r , $p_r \gg p_s$. Most peers

Assume:

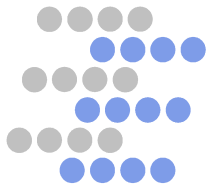
- Superpeers are stable peers

- Network delay inside groups is negligible (good locality)

Availability of peers:

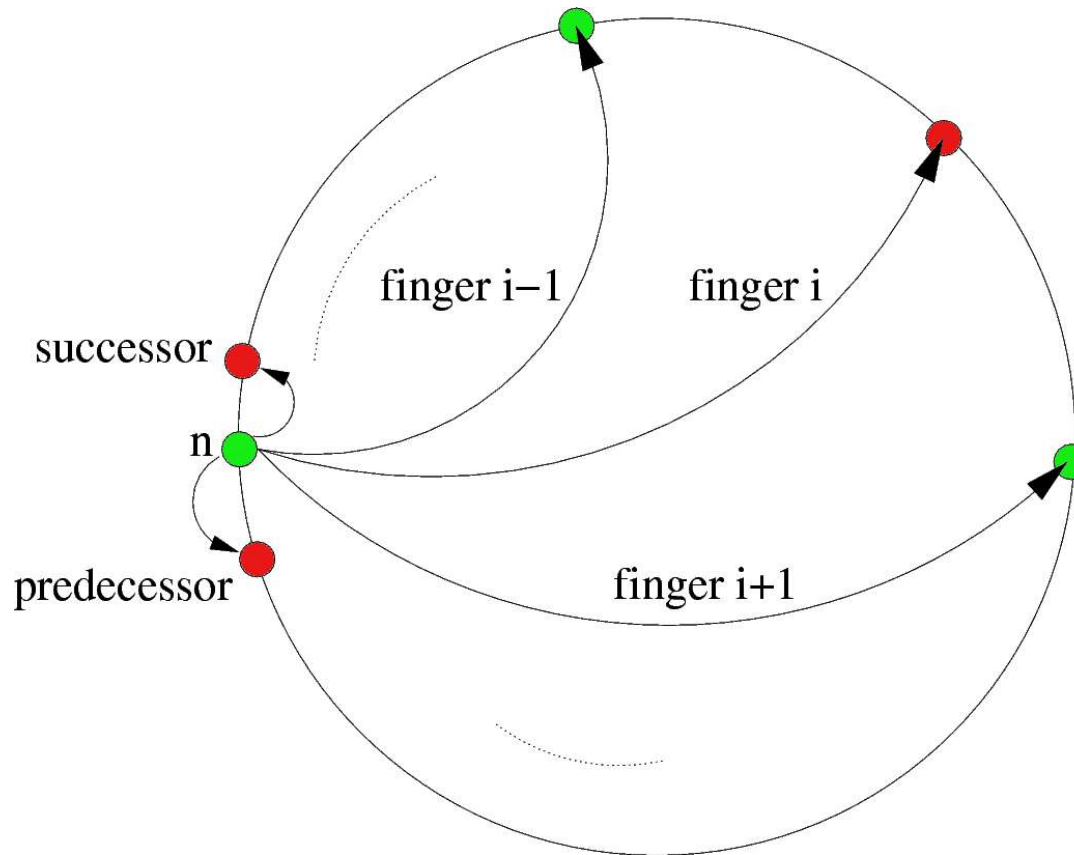
- In Chord: $1 - p_r$

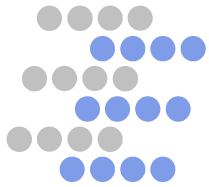
- In top-level Chord: $1 - p_s$ (or $1 - (p_s)^c$ for c superpeers per group)



Efficient Key Lookup (II)

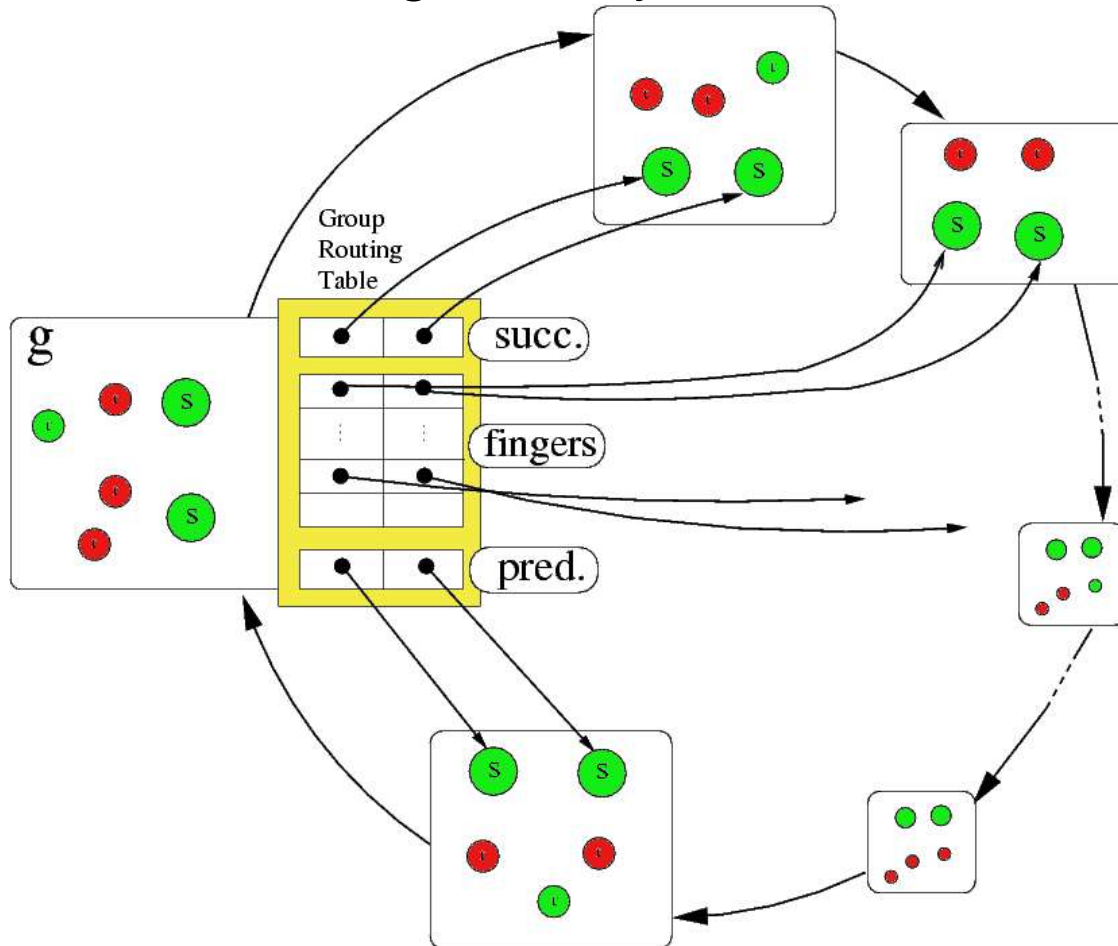
Chord lookup considering realistic peers

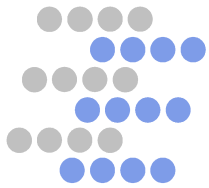




Efficient Key Lookup (III)

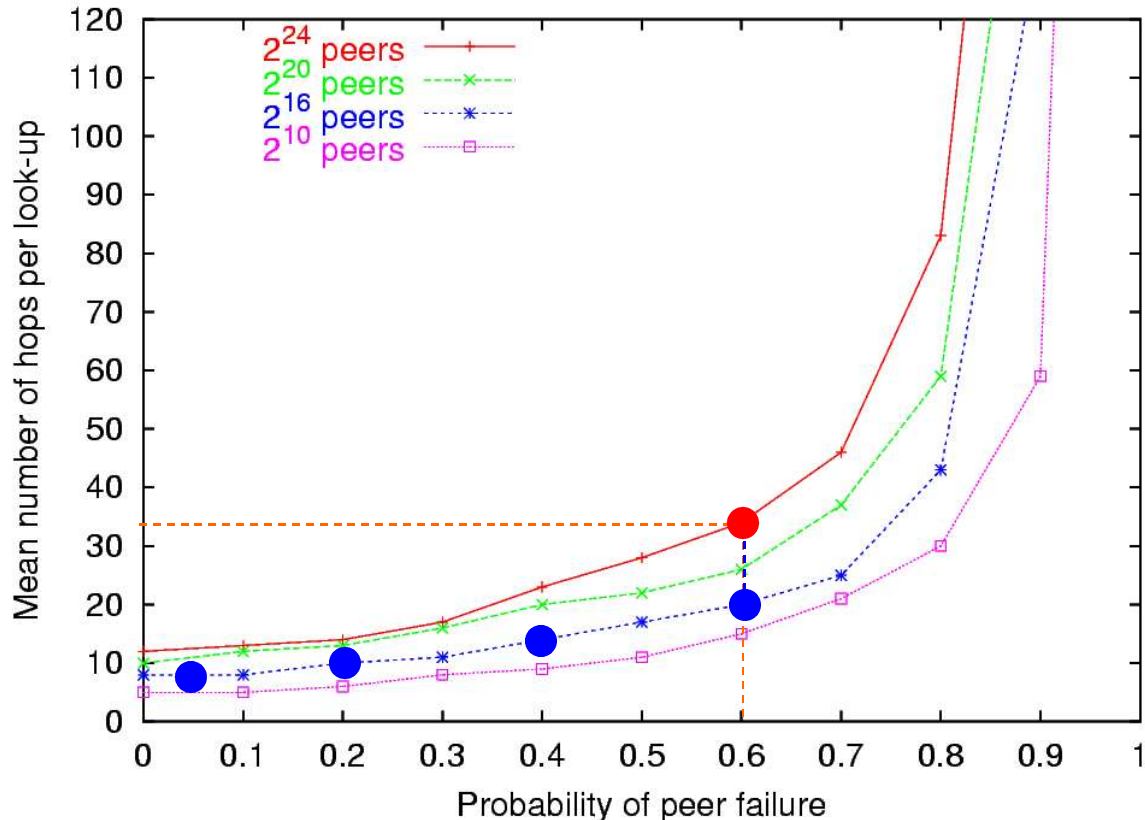
Top-level Chord ring stability



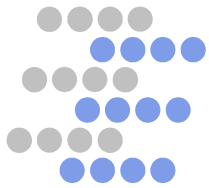


Efficient Key Lookup (IV)

Smaller and more reliable Chord ring gives more efficient key lookup



1. Flat Chord,
2²⁴ peers,
p=0.6
1. Top-level Chord,
groups of
2⁸=256 peers
1. Superpeers
p~0



Conclusion

Many Internet wide-deployed efficient systems use a hierarchical approach

We have presented **hierarchies of DHTs** in order to improve flat conventional DHTs:

- Exploit heterogeneous characteristics of peers

- Less traffic in the wide-area

- Caching infrastructure

- Transparency and autonomy of different parts of the system

- More efficient key lookup