Architectures and protocols in Peer-to-Peer networks

Ing. Michele Amoretti
[amoretti@ce.unipr.it]

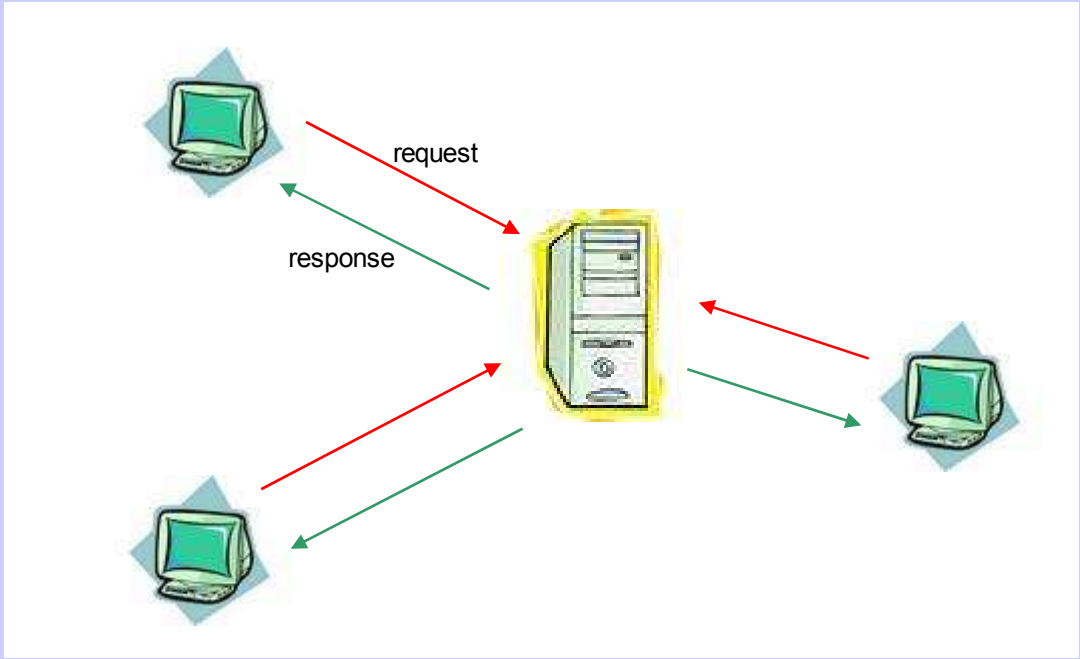
II INFN SECURITY WORKSHOP
Parma 24-25 February 2004

Contents

- Definition of Peer-to-Peer network
- P2P applications
- Taxonomies for P2P architectures
- P2P discovery algorithms
- P2P most important protocols
  - Napster and OpenNap
  - MFTP
  - BitTorrent
  - Direct Connect
  - Gnutella
  - FastTrack
  - Freenet
  - Chord

Definition of Peer-to-Peer network

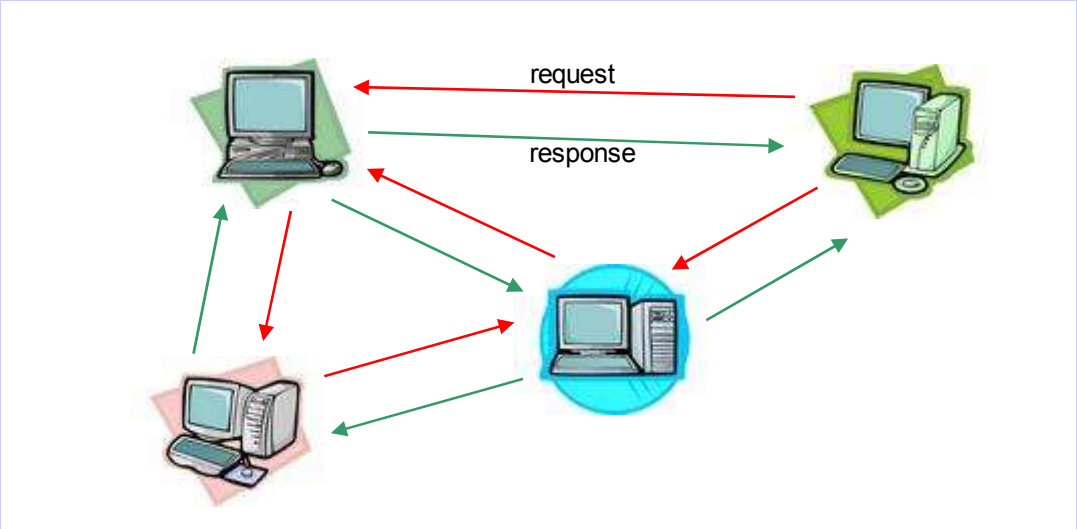
Client/Server network



In a C/S network each node or process is either a client or a server. Typically clients are lightweight, and rely on servers for resources and processing power.

Peer-to-Peer network

In a P2P network each node has both client and server functionalities, and can be partially or fully autonomous in the sense that it does not depend on a central authority.



P2P architectures are more scalable and robust than centralized systems which serve many clients bearing the majority of the cost of computation.

Peer-to-Peer applications

Parallelizable applications (distributed computing)

A large task is splitted in smaller sub-pieces that can execute in parallel over
a number of independent peer nodes.

Most often, the same task is performed on each peer using a different set of parameters (compute-intensive applications).

Examples: code breaking, market evaluation, scientific simulations.

Content and file management applications

The focus is on storing and retrieving information. In this field, P2P has been a disruptive technology (i.e. simple and cheap, but providing low profit margins and for this reason shunned by well-managed companies, which have been later damaged by P2P).

For the most part current implementations have not focused much on providing reliability and rely on the user to make intelligent choices about the location from which to fetch files and to retry when downloads fail.

Filtering and mining applications are beginning to emerge.

Collaborative applications

Users are allowed to collaborate, in real-time, without relying on a central server to collect and relay information.

- Instant messaging

- Applications that allow people to interact while viewing and editing the same information simultaneously

- Online games (MMFPS, MMORPG)

e.g. P2P collaboration with *Groove*

Groove Networks' collaboration software has already been licensed to over 10,000 employees at GlaxoSmithKline PLC, in the U.K., and is currently being tested for deployment in the U.S. by Raytheon Company and Abbott Laboratories.

Key features include:
- live voice over the Internet
- instant messaging
- threaded discussion
- content distribution tools for sharing files, pictures and contacts.

Users also have joint activity tools for simultaneous Web browsing and document editing, a white board for brainstorming, and a group calendar.

The *real* P2P killer application: Community Support

P2P systems could be community support system (community platform) that
provides a rich communication medium for work or interest groups.
Sociological analysis helps to characterize requirements which are new to distributed systems..

Targets:
University Campus, Research Labs, Enterprise, Finance

Open issues:

1 - coping with intermittent connectivity and presence

2 - lightweight protocols

3 - robustness, security (AAA)

In search for a common, open infrastructure: Project JXTA

The JXTA protocols define the minimum required semantic for peers to form and join an overlay network on top of the Internet.

Project JXTA is designed to be independent of programming languages, system platforms, service definitions and network protocols.

Peer
PeerGroups
Pipes
Services
…

} described by *advertisements*

Peers: *edge*, *RendezVous*, *Relay*

In search for a common, open infrastructure: Project JXTA - (2)

Core Specification Protocols (lower level) :

- Endpoint Routing Protocol (*ERP*)
- Peer Resolver Protocol (*PRP*)

Standard Service Protocols (higher level) :

- Peer Discovery Protocol (*PDP*)
- Peer Information Protocol (*PIP*)
- Pipe Binding Protocol (*PBP*)
- RendezVous Protocol (*RVP*)

Taxonomies for P2P architectures

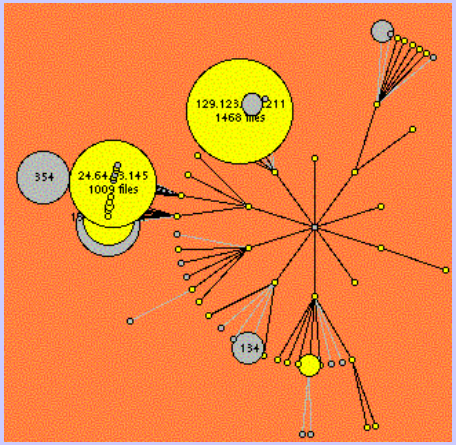
Pure vs *hybrid* P2P networks

The term *pure P2P computing* refers to an environment where all the participanting nodes are peers. No central system controls, coordinates, or facilitates the exchanges among the peers.

In *hybrid P2P computing* there are servers which enable peers to interact with each other. The degree of central system involvement varies with the application.

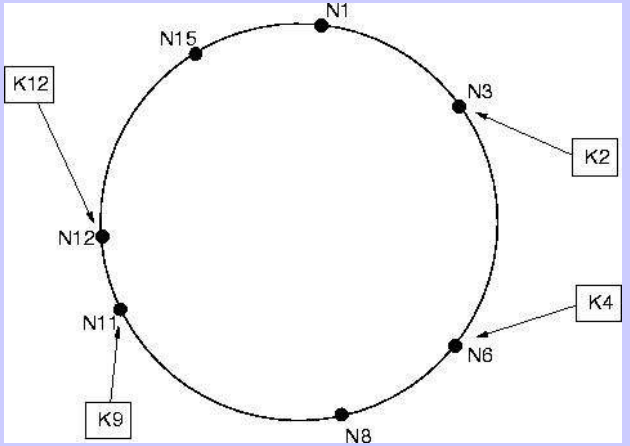
No one method is better than the other: each has its advantages and its drawbacks, each is the right choice for some applications.

Unstructured vs *structured* P2P networks [Barkai 2001]



Unstructured overlay networks (based on protocols such as Napster, Gnutella, Freenet) are not embedded with a logically deterministic structure for organizing and managing the peer nodes.

Structured overlay networks (based on protocols such as Chord, CAN, Tapestry, Tornado) manage the peer nodes with an implicit logical and deterministic structure.



2) *Five-dimensional technological classification* [Kant et al. 2002]

Resource Storage (data):
from *organized* (res. located in globally known nodes) to *scattered*.

Resource Control (metadata):
from *organized* (informations located in globally known nodes) to *scattered*.

Resource Usage:
from *isolated* to *collaborative* (multicasting, RPC, call backs, ..).

Global state control:
from loose to tight.

QoS constraints:
from loose (non real-time) to tight (e.g. streaming media).

P2P discovery algorithms

Centralized Directory Model (CDM)

The peers connect to a central directory where they publish informations about the content they offer for sharing.

Upon request from a peer, the central index will find the best peer that matches the request.

Advantages: simple, high degree of control on shared contents.
Limits: not scalable, single point of failure.

E.g.: Napster, Direct Connect

Flooded Requests Model (FRM)

Pure P2P algorithm in which each request from a peer is flooded (broadcasted) to directly connected peers, which themselves flood their peers, etc.

Advantages: efficient in limited communities (i.e. not very scalable).
Limits: requires large bandwidth.

E.g.: Gnutella, FastTrack

Document Routing Model (DRM)

This algorithm is based on Distributed Hash Tables (DHT).

Publishing of a document:
routing it to the peer whose ID is the most similar to the document ID, and repeating the process until the nearest peer ID is the current peer's ID.

Discovery:
the request goes to the peer whose ID is the most similar to the document ID, and the process is repeated until the document is found.

Advantages: scalable.
Limits: malicious participants can threaten the liveness of the system.

E.g.: FreeNet, Chord, CAN, Tapestry.

P2P most important protocols

Napster and OpenNap

Network architecture: hybrid, unstructured
Algorithm: CDM

With Napster, the files stays on the client machine, never passing through the servers. The servers provide the ability to search for particular files and initiate directs transfers beetween clients.

OpenNap extends the Napster protocol to allow sharing of any media type and the ability to link servers togheter.

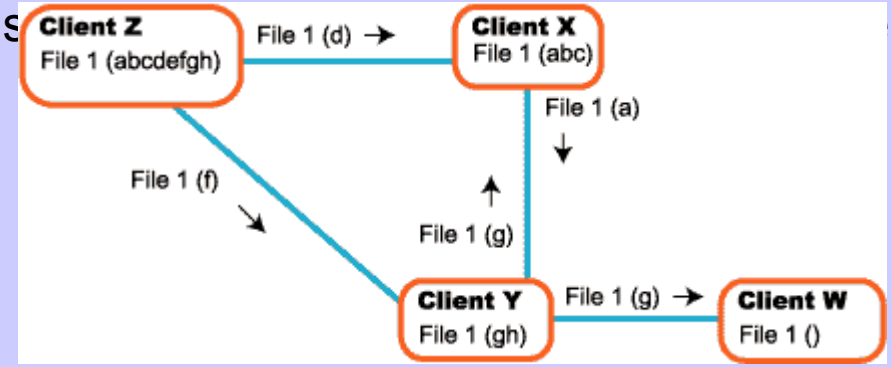
Napster 2.0 now offers online music store services, delivering access to the largest catalog of online music (more than 500.000 tracks).

http://www.napster.com
http://www.winmx.com

Multisource File Transfer Protocol (MFTP)

Network architecture: hybrid or pure, unstructured
Algorithm: CDM or DRM

MFTP is designed to spread files in the fastest possible way beetween many interested peers. To achieve this purpose peers download files from several sources concurrently, i.e. a downloader becomes a source to other downloaders as s.



This approach is based on the Tit For Tat strategy, which is usally applied to the repeated Prisoner's Dilemma [Tucker 1950].

Multisource File Transfer Protocol (MFTP) - (2)

The most important overlay network based on MFTP is eDonkey2000, which
is based on the CDM discovery algorithm. Each peer publish informations about its contents to servers that can be set up by anyone.

Once the network reaches a certain size, the servers become a bottleneck to the performance, and users can no longer search the entire network for things they are interested in (a sort of unwanted *clustering*).

The most recent overlay network based on MFTP, Overnet, uses the DRM algorithm to decentralize searching and publishing. Each node in the network knows about a small set of other nodes, and data are organized using a DHT.

http://www.edonkey2000.com

BitTorrent

Network architecture: hybrid, unstructured
Algorithm: CDM

The (Web) servers don't have informations about content location. They only
store metainfo files describing the objects (length, name, etc.) and associating to each of them the URL of a *tracker*.

Trackers are responsible for helping downloaders find each other, speaking a very simple protocol layeres on top of HTTP.
Moreover, a downloader sends status info to trackers, which reply with lists of contact information for peers which are downloading the same file.

BitTorrent cuts files into pieces, which are broken into sub-pieces. Pieces are propagated with the same strategy used by MFTP (based on TFT).

http://bitconjurer.org/BitTorrent/

Direct Connect

Network architecture: hybrid, unstructured
Algorithm: CDM

The DC network is composed of Hubs, Clients, and by the HubListServer.

Hubs act as naming services and communication facilitators for Clients, allowing them to exhange search commands and chat messages.

The HubListServer acts as a naming service: Clients discover Hubs asking the HLS.

Official DC Client: http://www.neo-modus.com
DC++: http://dcplusplus.sourceforge.net

Gnutella

Network architecture: pure, unstructured
Algorithm: FRM

A Gnutella node connects to the network by reaching one of the several known hosts which are almost always available.

The messages allowed in the Gnutella network can be grouped as follows:

Group Membership (`PING` and `PONG`, for peer discovery queries/replies)
Search (`QUERY` and `QUERY HIT`, for file discovery queries/replies)
File Transfer (`GET` and `PUSH`, for file exchange between peers)

To avoid network congestion, the PING and QUERY message are always associated to a Time To Live (TTL):

$$TTL(0) = TTL(i) + Hops(i)$$

Gnutella - (2)

The cost of flooding-based broadcast, expressed as number of messages forwarded:

$$c = \sum_i m_i = 1 + \sum_i (d_i - 1) = 1 + N(d - 1)$$

where N is the total number of nodes in the network, and d is the average node degree.

Assuming that all N nodes initiate broadcasts with the same constant rate r, the average bandwidth usage per node is:

$$B = B_{tot} / N = (2cNr) / N = 2cr$$

In a network with N = 10000, d = 4 and r = 1/min we have B = 176Kbps.

Gnutella - (3)

In the real Gnutella network it has been observed that the high cost of broadcasting and the lack of resources of a large number of participants lead to fragmentation of the network into smaller subnetworks.

Gnutella node connectivity: *multi-model distribution, combining a power law and a quasi-constant distribution*.

$P(d) \sim d^{-\gamma}$

Internet connectivity: *power law distribution*.

The mismatch between the two topologies leads to
- ineffective use of the physical networking infrastructure
- some lack of robustness

but also to
- quite good fault tolerance

(pure power laws networks are strongly affected by the removal of one hub).

FastTrack

Network architecture: hybrid, unstructured
Algorithm: FRM

The FT protocol is an extension of the Gnutella protocol which adds supernodes to improve scalability.

A peer application hosted by a powerful machine with a fast network connection become automatically a supernode, effectively acting as a temporary indexing server for other slower peers.

The supernodes communicate between each others in order to satisfy search requests.

KaZaA Media Desktop: http://www.kazaa.com
Grokster: http://www.grokster.com
iMesh: http://www.imesh.com

FreeNet

Network architecture: pure, unstructured
Algorithm: DRM

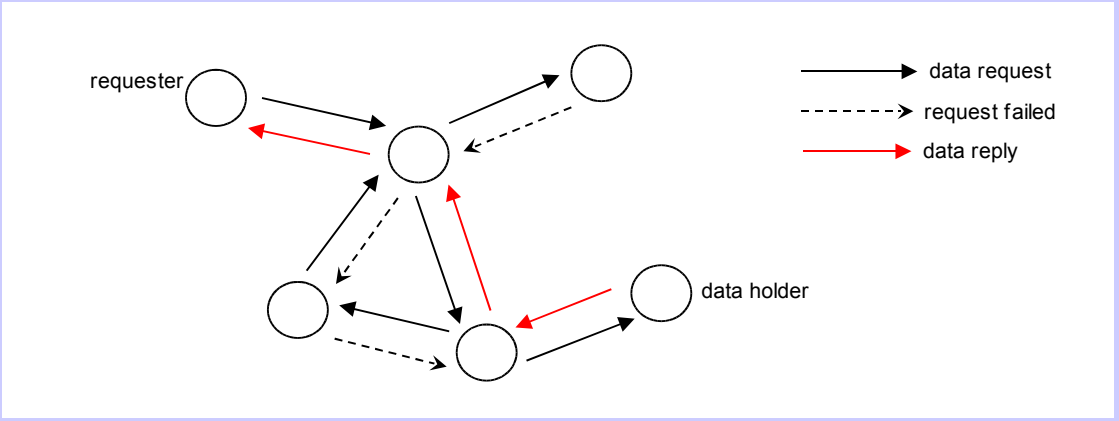
Focus on:

- privacy for information producers, consumers and holders
- resistance to information censorship
- high availability and reliability through decentralization
- efficient, scalable and adaptative storage routing

When a peer wants to share a file, it uses an hash function (typically SHA-1)
to generate a key from a text description of the file.

Every node maintains a routing table that lists the addresses of other nodes and the files they holds (with high probability).

FreeNet - (2)



When a node receives a query it first check its own store and, if it finds the key, returns the file.

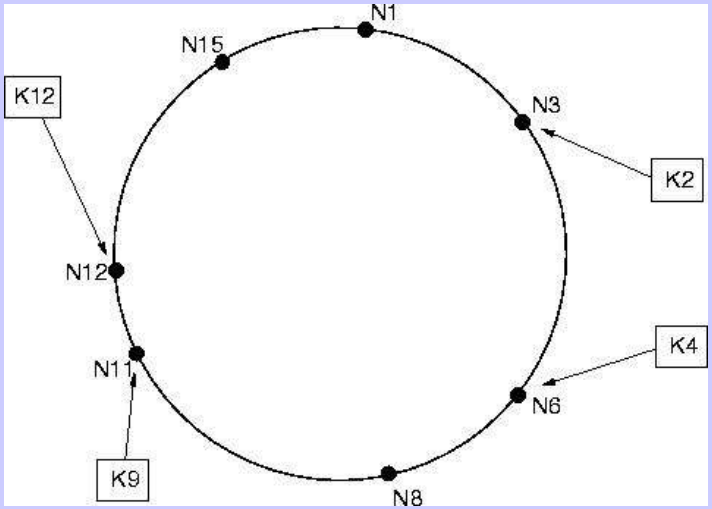
Otherwise, the node forwards the request to the node in its table with the closest key to the one requested (Freenet attempts to cluster files with similar keys).

When the file is found, each node in the chain passes the file back upstream and creates a new entry in its routing table. Depending on the distance from the holder, each node might also cache a copy locally.

Chord

Network architecture: pure, structured
Algorithm: DRM

An hash function (such as SHA-1) is used to assign each node and key (identifying a file) an *m*-bit identifier.
Node identifiers are ordered on an identifier circle modulo 2^m.



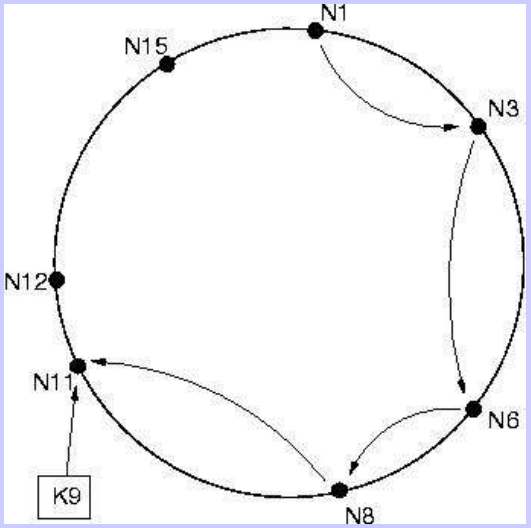
Key *k* is assigned to the first node whose identifier is equal or follows the identifier of *k*.
This node is called the *successor*

of key *k*.

Chord - (2)

Basic lookup algorithm:

Queries for a given key identifier are passed around the circle via successor pointers until they encounter a pair of nodes that straddle the desired Identifier; the second in the pair is the node the query maps to.



The result returns along the reverse of the path followed by the query.

The number of nodes that must be

contacted to find a successor in an *N*-node network is

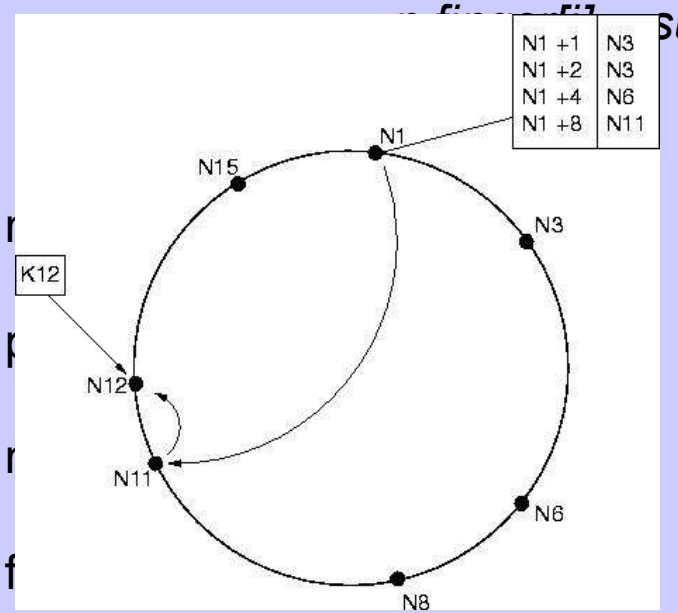
O(N)

Chord - (3)

Accelerated lookup algorithm:

Each node *n* maintains a routing table with up to *m* entries, called the finger table. The i^{th} entry is:

successor$(n + 2^{i-1})$



Using the finger table to find the

whose identifier immediately

the one of the desired key, the

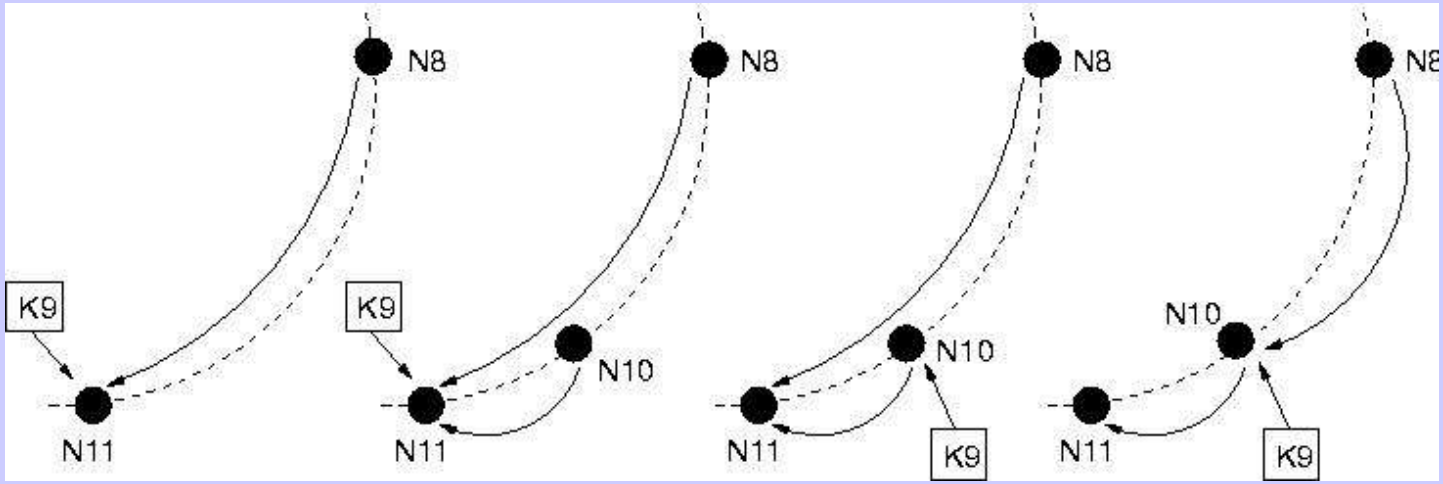
of nodes that must be contacted to

a successor in an *N*-node network is

O(logN)

Chord - (4)

In order to ensure that lookups execute correctly as the set of participating nodes changes, Chord introduces a stabilization protocol that each node should run periodically in the background to update finger tables and successor pointers.



Chord - (5)

The correctness of the Chord protocol relies on the fact that each node knows its successor. Howerver, this invariant can be compromised if nodes fail.

To increase robustness, each Chord node maintains a *successor list* containing the node's first *r* successors.

A typical application using Chord might store replicas of the data associated

with a key to the $l \leq r$.

THANK YOU!

DHT

Basically, a DHT performs the function of a hash table.

You can store a key and a value pair.
You can lookup a value if you have the key.

The interesting thing about DHTs is that storage and lookups are distributed among multiple nodes.

Using a hash to generate the key from the value is beneficial because hashes generally are distributed evenly, and different keys are distributed evenly across all the nodes in the network.

Tipically, the used hash function is based on SHA-1.

Secure Hash Algorithm (SHA-1)

SHA-1 computes a condensed representation of a message or a data file. When a message of any length < 2^{64} bits is input, the SHA-1 produces a 160-bit output called a message digest.

The message digest can then be input to the Digital Signature Algorithm (DSA) which generates or verifies the signature for the message. Signing the message digest rather than the message often improves the efficiency of the process because the message digest is usually much smaller in size than the message.

The same hash algorithm must be used by the verifier of a digital signature as was used by the creator of the digital signature.

Secure Hash Algorithm (SHA-1) - (2)