

Peer-to-Peer Computing: DHTs and Overlay Networks

Romaric Duvignau

Associate Professor, Chalmers University of Technology, Gothenburg, Sweden

<https://www.cse.chalmers.se/~duvignau/>

duvignau@chalmers.se

Today's lecture

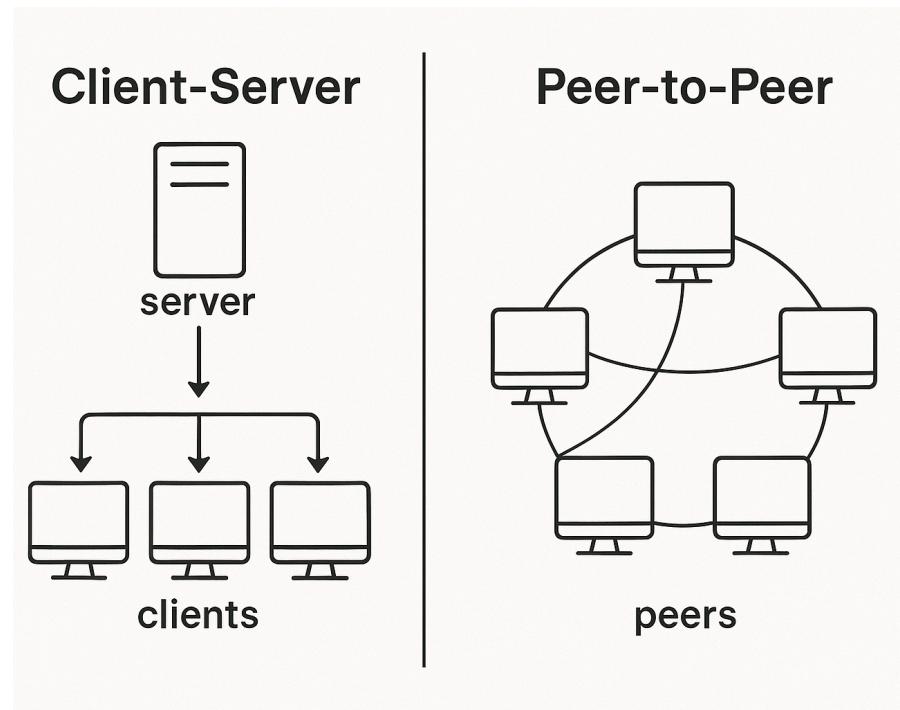
Example of unstructured overlays

- Napster 1999 vs. Gnutella 2000
- BitTorrent 2001

Overlay Network?

Structured Overlays / DHT

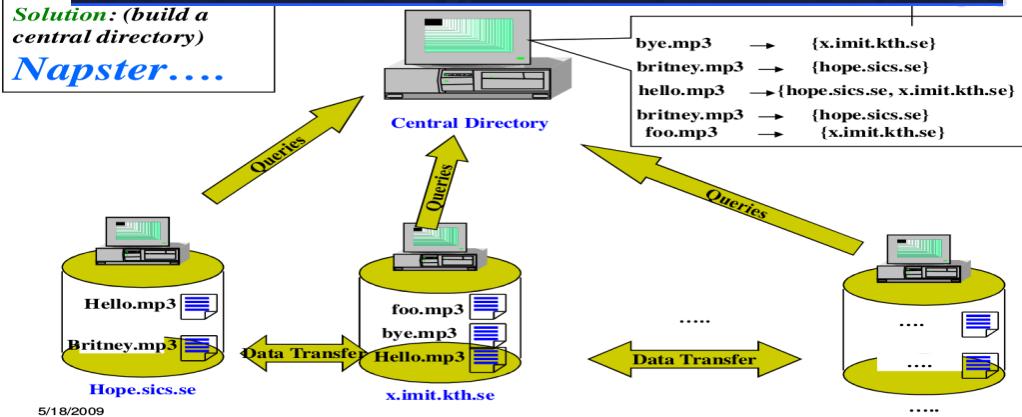
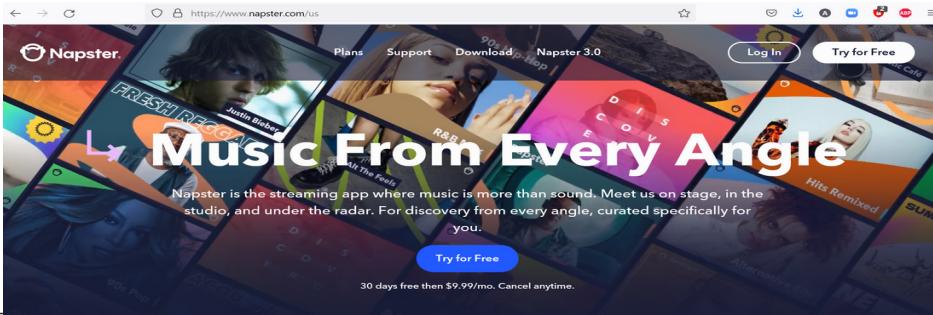
- How?
- Chord



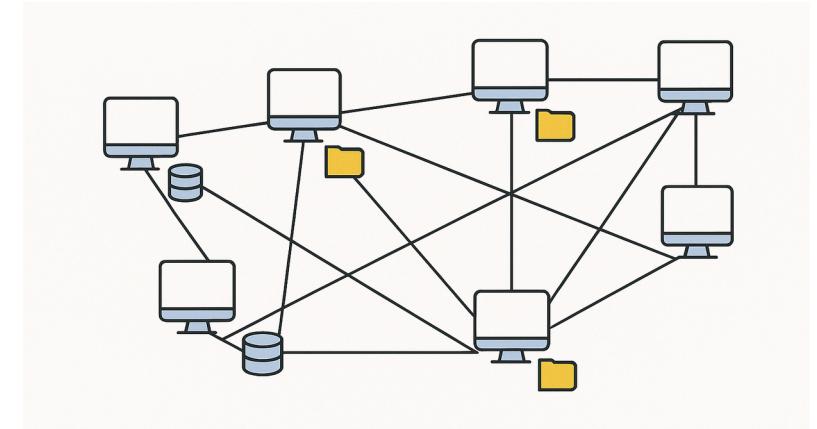
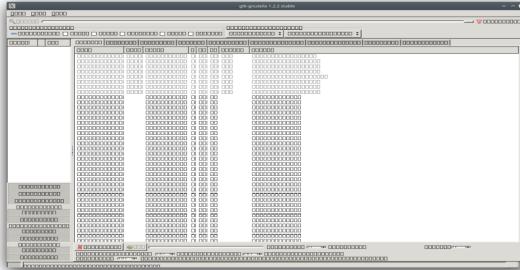
Unstructured overlays

Unstructured Overlays

Napster



Gnutella



straightforward



Can be plugged off



distributed



flooding

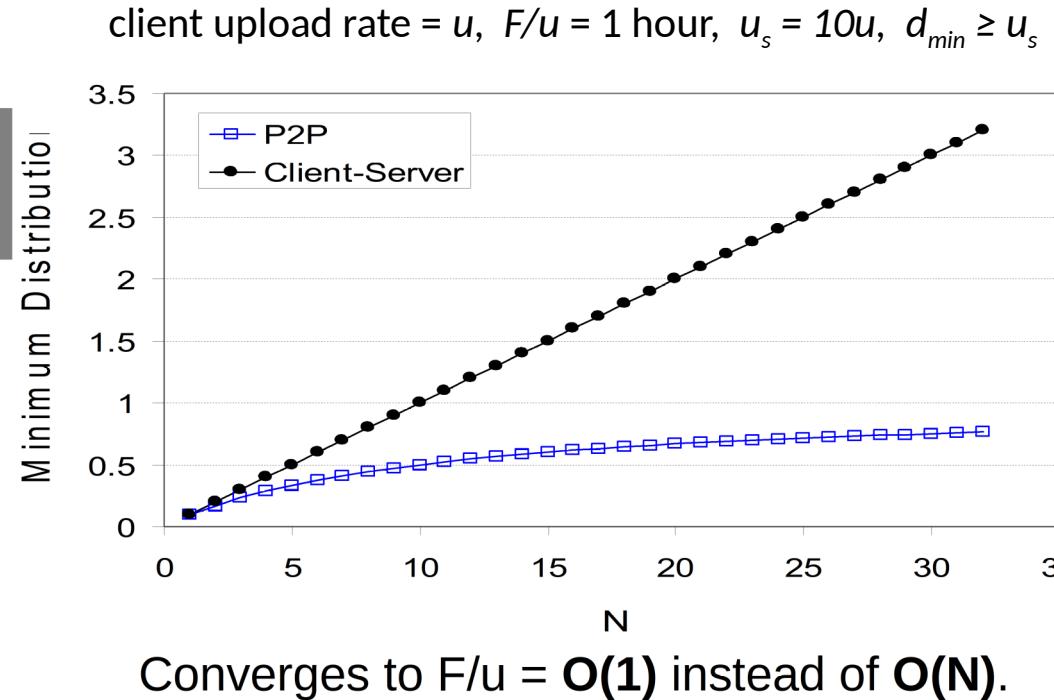
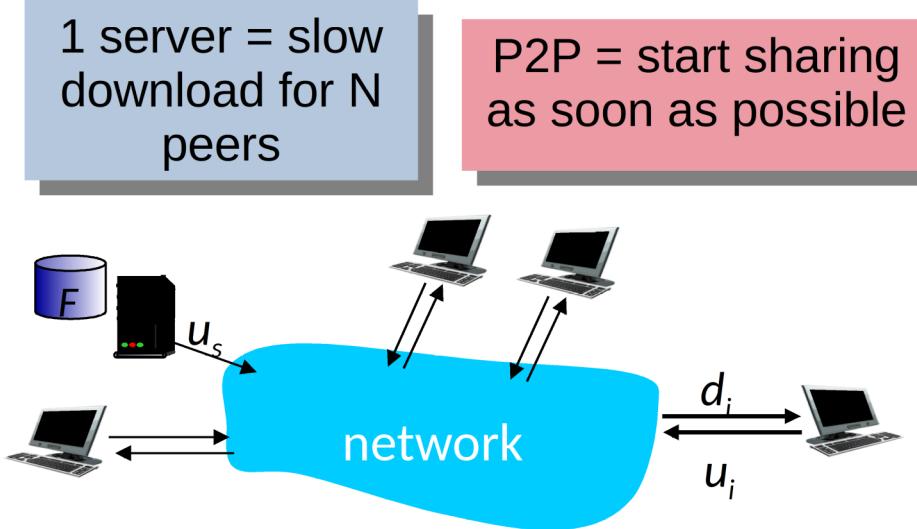
BitTorrent: Content distribution with a twist

- Large number of peers want to download a certain file as fast as possible.
- All previous techniques discussed **do not do that!**

Bittorrent's philosophy:
Use the bandwidth of everyone
in the network to the maximum!



The advantage of P2P for file sharing



time to distribute F to N clients using P2P approach

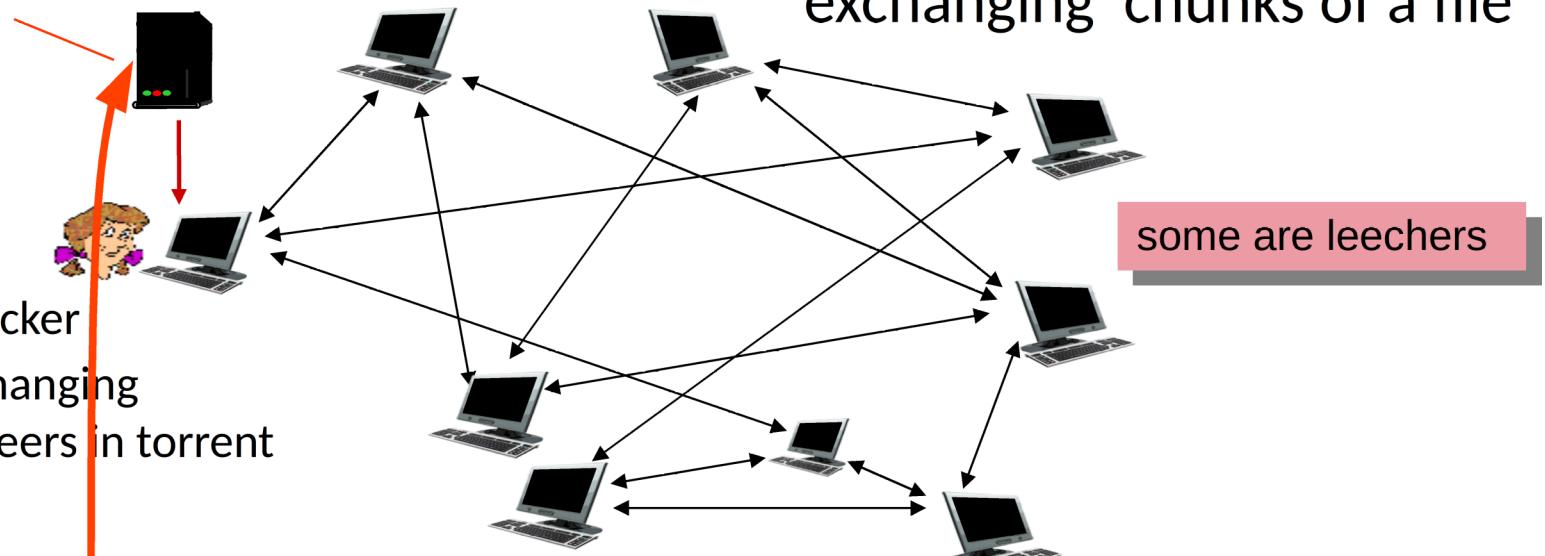
$$D_{P2P} > \max\{F/u_s, F/d_{min}, NF/(u_s + \sum u_i)\}$$

BitTorrent in action

tracker: tracks peers participating in torrent

Alice arrives ...
... obtains list
of peers from tracker
... and begins exchanging
file chunks with peers in torrent

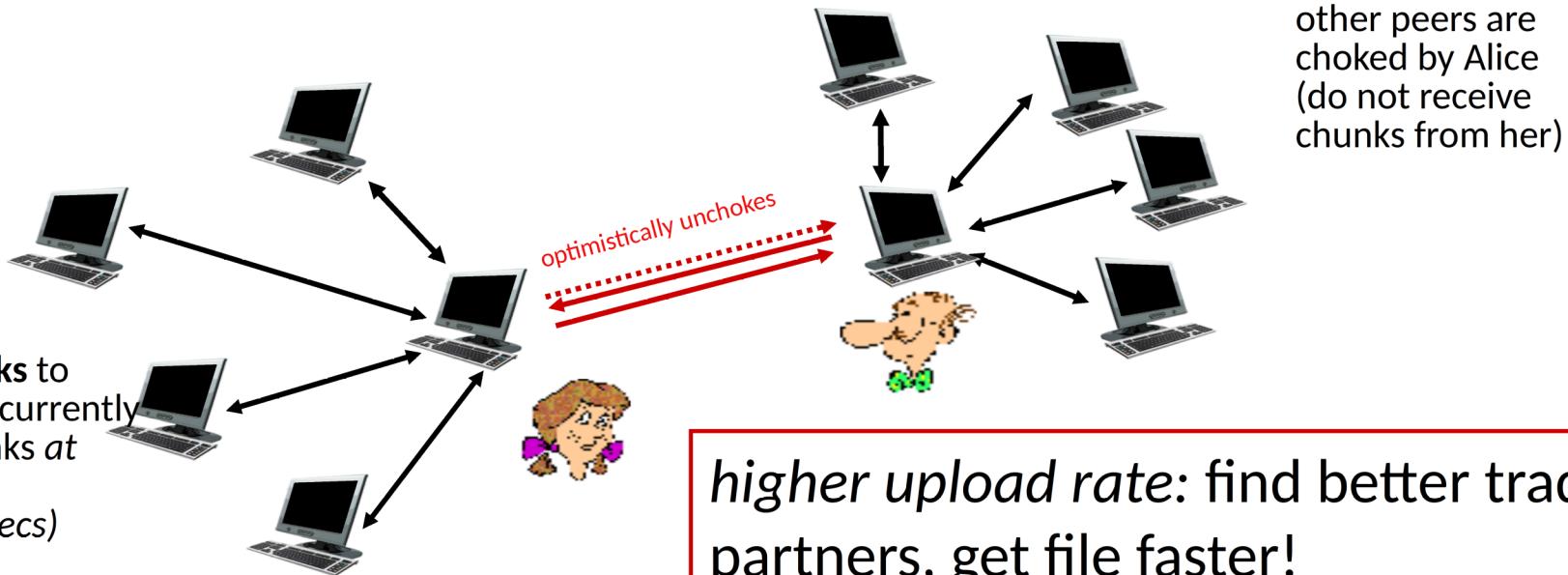
swarm: group of peers exchanging chunks of a file



```
Big_Buck_Bunny_1080p_s...frostwire.com.torrent ✘
H8:announce42:http://tracker.frostwire.com:6969/announce13:creation
datei123118572e4:infodl6:length192867054e4:path147:Big_Buck
E_YOUR_CONTENT_ON_FROSTWIRE_01_06_09.txtseed6:lengthi3456234e4:path139
e.txteee4:name58:Big_Buck_Bunny_1080p_surround_frostclick.com_frostwi
\[ ] & )q^ @m _ 0 aQ . S $ h ? 0~ b ^ F - 8w M3
V=iTh q i h qv w dH f? wM h > / V I V
n < F k H H a = R f n?F tR "+ n v i k n v t r + e
0 a >1 f) H ^ n H j L f=r , g X P n & " ä c U :
1 n i l l i K RM AR Q XF C G B 14 ' v s 1 > 2 m Q F4 i N
Pii > < K n R 3 f ~ ii n & R c T ' e / P C k 7 n f R 3 ' f F S 7 F 7
```

Choking algorithm: Mutual love or Tit-for-tat

Files are divided in **fixed-size chunks**,
downloaded in rarest-first/random/streaming order.



Reflections on BitTorrent (historical)



Hybrid P2P/centralized architecture, flexible and high performance!



Dependence on a centralized component for the search function!

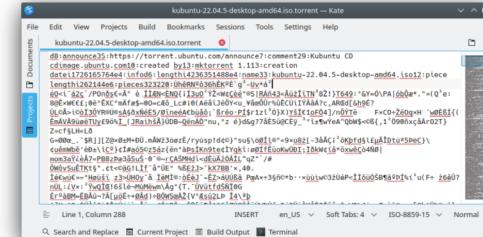


The screenshot shows the Transmission desktop application interface. At the top, there's a menu bar with File, Edit, Torrent, View, Help. Below it is a toolbar with Open, Properties, and other icons. A main window displays a single torrent entry: "ubuntu-22.04.5-desktop-amd64.iso" with a size of 1.76 GB and a download speed of 5.54 MB/s. A progress bar indicates the download is at 4 minutes remaining. Below the main window is a detailed view of the torrent's properties, including its SHA256SUMS file.

| Name | Last modified | Size | Description |
|---|------------------|------|---|
| Parent Directory | | - | |
| SHA256SUMS | 2024-09-12 18:29 | 100 | |
| SHA256SUMS.gpg | 2024-09-12 18:29 | 833 | |
| kubuntu-22.04.5-desktop-amd64.iso | 2024-09-11 14:48 | 3.9G | Desktop image for 64-bit PC (AMD64) computers (standard download) |
| kubuntu-22.04.5-desktop-amd64.iso.torrent | 2024-09-12 18:29 | 316K | Desktop image for 64-bit PC (AMD64) computers (BitTorrent download) |
| kubuntu-22.04.5-desktop-amd64.zsync | 2024-09-12 18:29 | 7.9M | Desktop image for 64-bit PC (AMD64) computers (zsync metafile) |
| kubuntu-22.04.5-desktop-amd64.list | 2024-09-11 14:48 | 13K | Desktop image for 64-bit PC (AMD64) computers (file listing) |
| kubuntu-22.04.5-desktop-amd64.manifest | 2024-09-11 14:36 | 64K | Desktop image for 64-bit PC (AMD64) computers (contents of live filesystem) |

This screenshot shows a web browser displaying the download page for the Kubuntu 22.04.5 LTS desktop image. The URL is "ubuntu-22.04.5-desktop-amd64.iso". The page includes a "Desktop image" section with a warning about RAM requirements and a "Available files" section listing various torrent files and their details. A red arrow points from the "kubuntu-22.04.5-desktop-amd64.iso.torrent" entry in the table below to the "kubuntu-22.04.5-desktop-amd64.iso" file in the browser's download manager.

| Name | Last modified | Size | Description |
|---|------------------|------|---|
| Parent Directory | | - | |
| SHA256SUMS | 2024-09-12 18:29 | 100 | |
| SHA256SUMS.gpg | 2024-09-12 18:29 | 833 | |
| kubuntu-22.04.5-desktop-amd64.iso | 2024-09-11 14:48 | 3.9G | Desktop image for 64-bit PC (AMD64) computers (standard download) |
| kubuntu-22.04.5-desktop-amd64.iso.torrent | 2024-09-12 18:29 | 316K | Desktop image for 64-bit PC (AMD64) computers (BitTorrent download) |
| kubuntu-22.04.5-desktop-amd64.zsync | 2024-09-12 18:29 | 7.9M | Desktop image for 64-bit PC (AMD64) computers (zsync metafile) |
| kubuntu-22.04.5-desktop-amd64.list | 2024-09-11 14:48 | 13K | Desktop image for 64-bit PC (AMD64) computers (file listing) |
| kubuntu-22.04.5-desktop-amd64.manifest | 2024-09-11 14:36 | 64K | Desktop image for 64-bit PC (AMD64) computers (contents of live filesystem) |

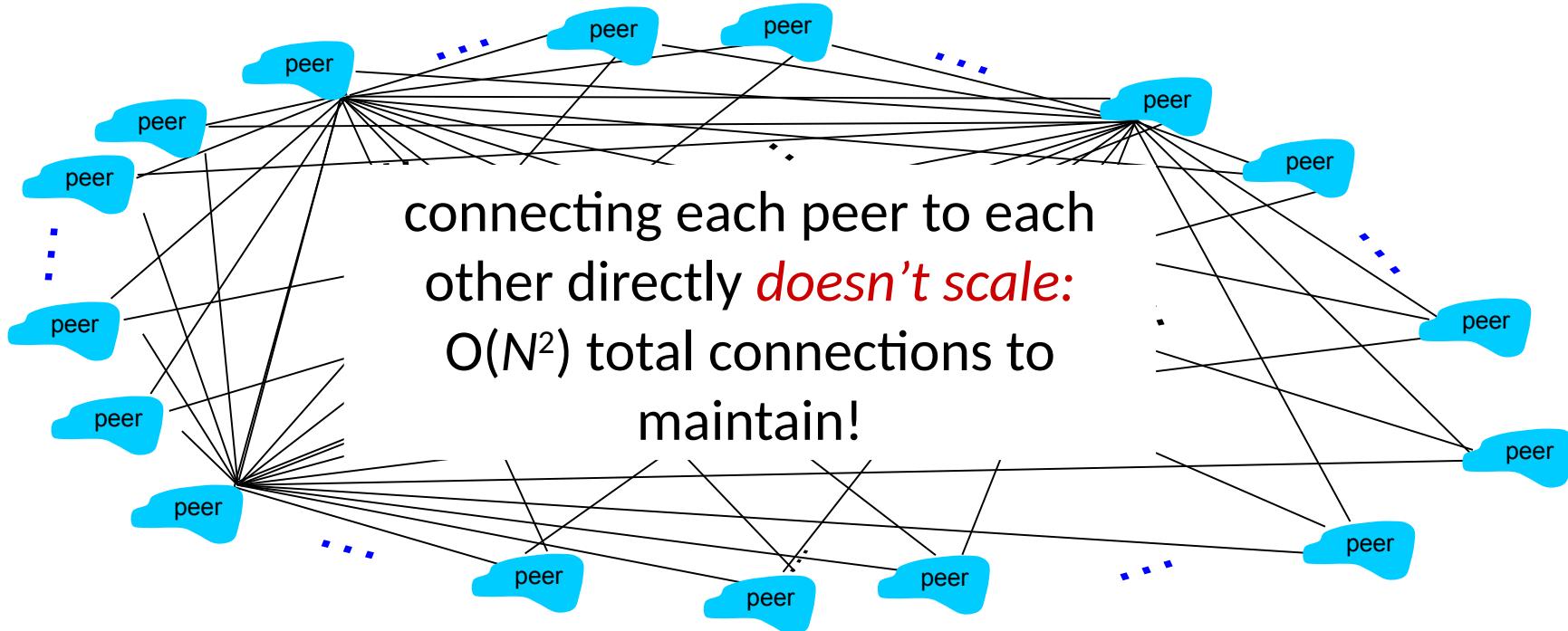


This screenshot shows a file manager window with two entries: "kubuntu-22.04.5-desktop-amd64.iso" and "kubuntu-22.04.5-desktop-amd64.torrent". The "kubuntu-22.04.5-desktop-amd64.iso" file has a status message "9m left — 161 MB of 3.9 GB (6.7 MB/sec)". The "kubuntu-22.04.5-desktop-amd64.torrent" file has a status message "File moved or missing".

The rise of DHTs!

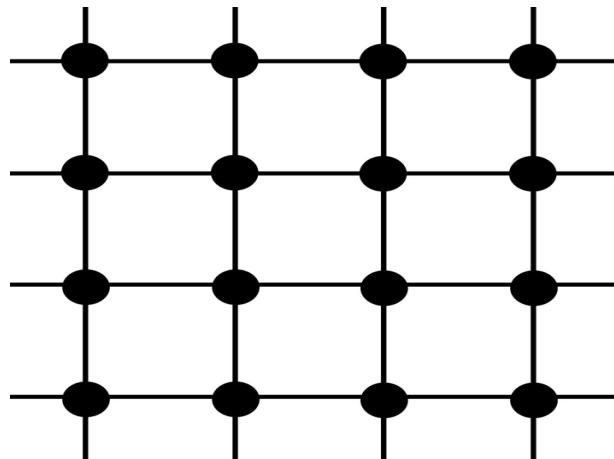
Structured Overlays

How to structure a P2P network?



Can we do better?

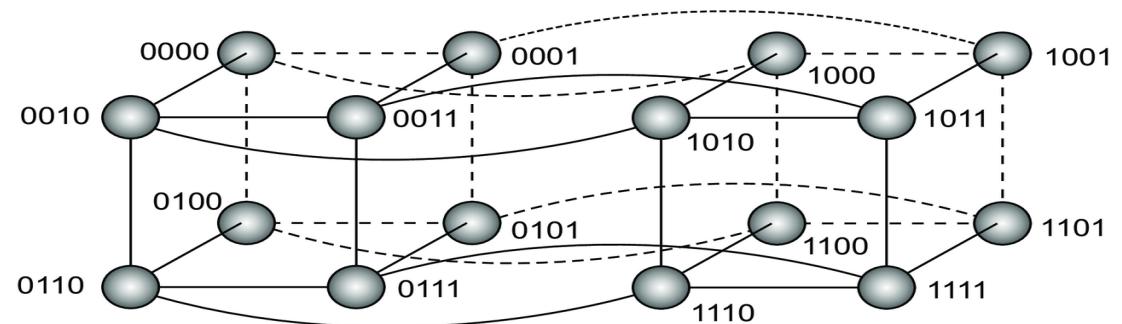
Lattice Graph



Degree: Constant!

Diameter: Linear!

Hypercube

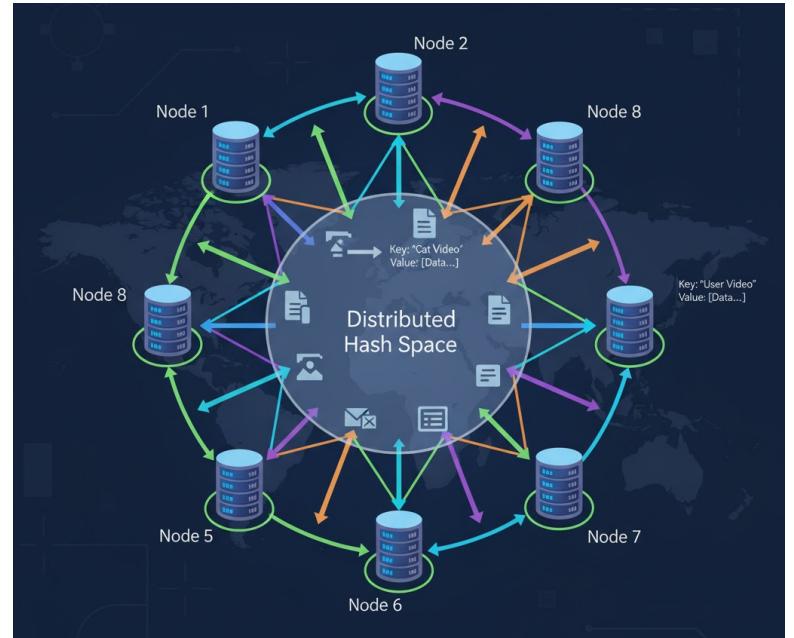
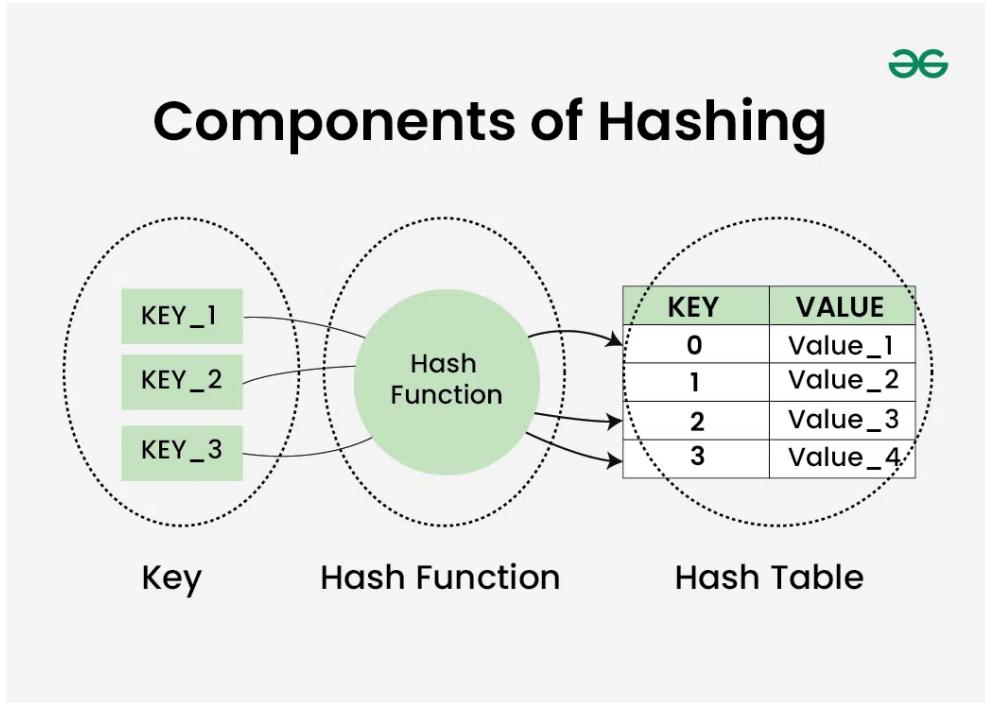


Degree: Logarithmic!

Diameter: Logarithmic!

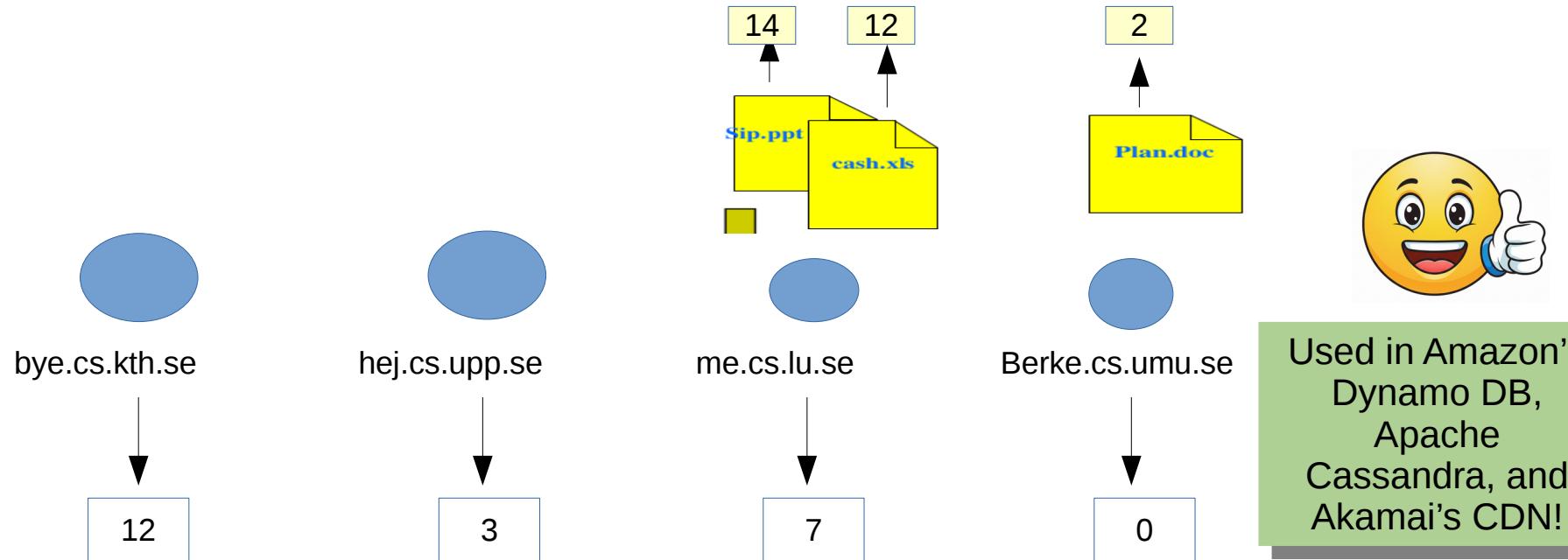
Chord (2001)
has similar
properties to an
hypercube!

What is a DHT?



For example, **BitTorrent's DHT** (based on Kademlia) stores **infohash** → <peer list>

Simplest Example: Consistent Hashing

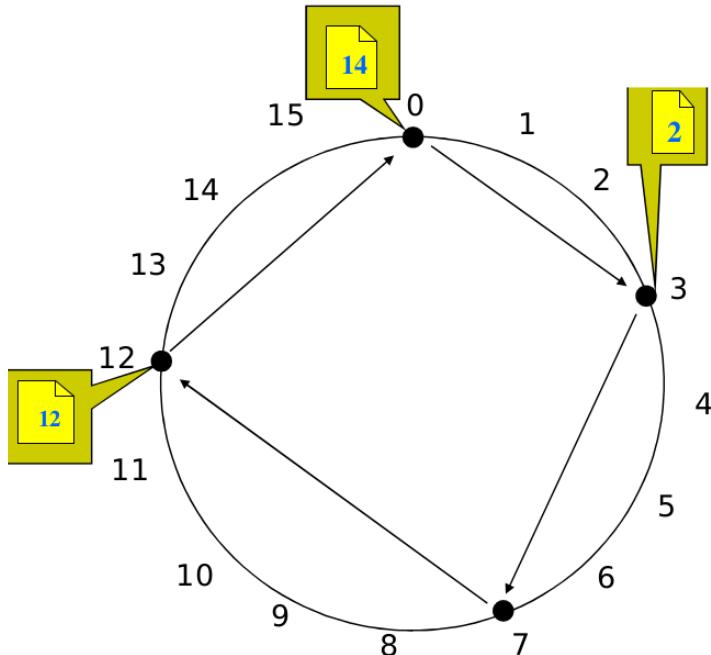


Used in Amazon's
Dynamo DB,
Apache
Cassandra, and
Akamai's CDN!

- 4 machines that need to share data
- Each machine will have an id based on the hash of its IP address
- Data will also have ids using the same function (SHA, MD5sum..)

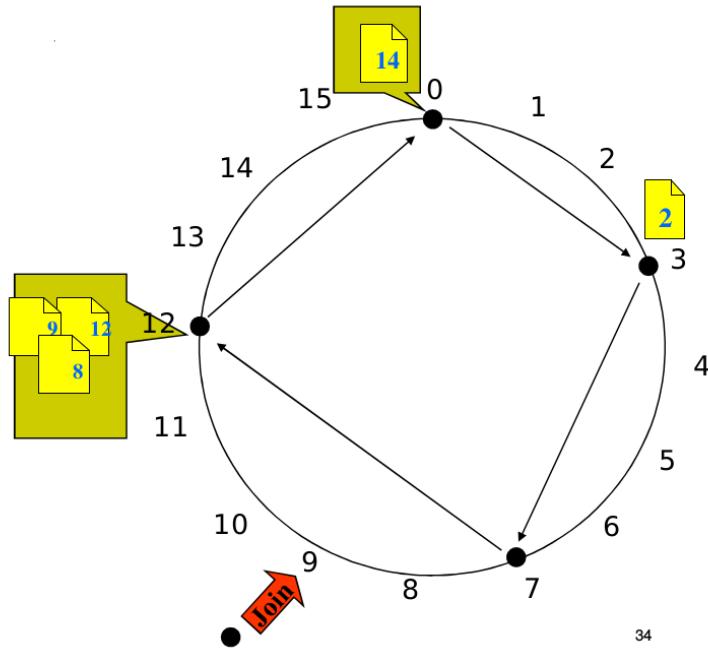
The **ID** of a data item determines the machine on which it is going to be stored.

Simple Example: build and update the ring



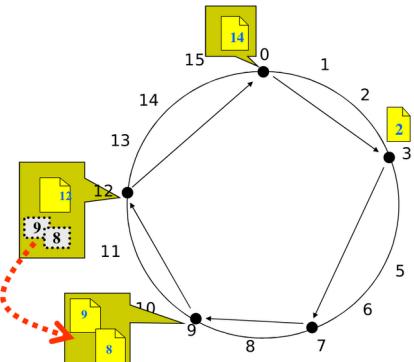
- Policy
 - A doc with **id y**, would be stored at **Succ(y)**
- Lookup
 - ask **node n** which you know about to find the successor of **id**.

New example: Handling Joins



- You need to know the successor
- Your predecessor needs to know about you
- Example: A **node 9** joins the ring

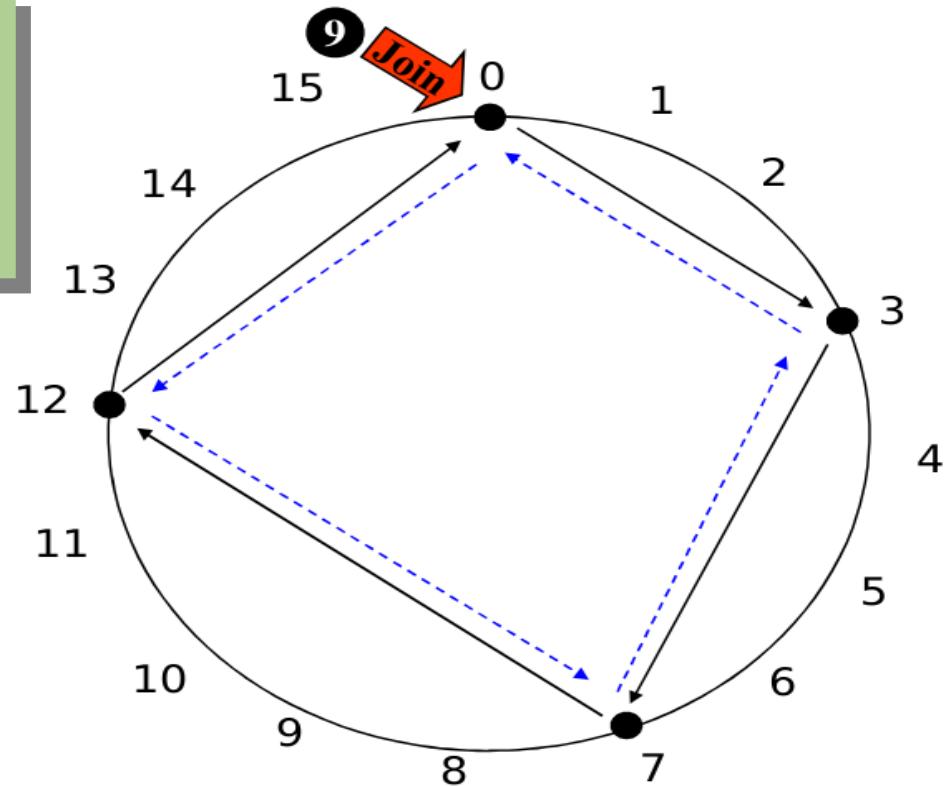
Can you think what happens to data?



Chord & Chord's Join

- Developed at MIT, published in 2001
- Very popular since it is very simple to understand
- In addition to the successor pointer, every node has a **predecessor pointer** as well

- 9 can join through any other node, take 0 for example.
- 9 will set its predecessor to nil
- 0 will help 9 to find its successor
- 0 will tell 9 that its successor should be 12



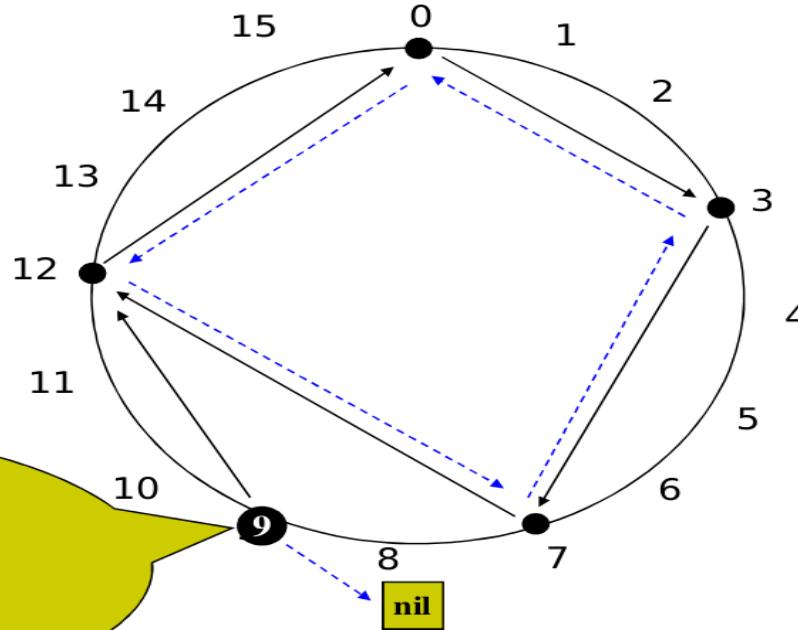
Chord Join Cont'd

```
// called periodically. verifies n's immediate  
// successor; and tells the successor about n.  
n.stabilize()
```

```
x = successor.predecessor;  
if (x ∈ (n, successor))  
    successor = x;  
successor.notify(n);
```

```
// n' thinks it might be our predecessor:  
n.notify(n')  
if (predecessor is nil or n' ∈ (predecessor, n))  
    predecessor = n';
```

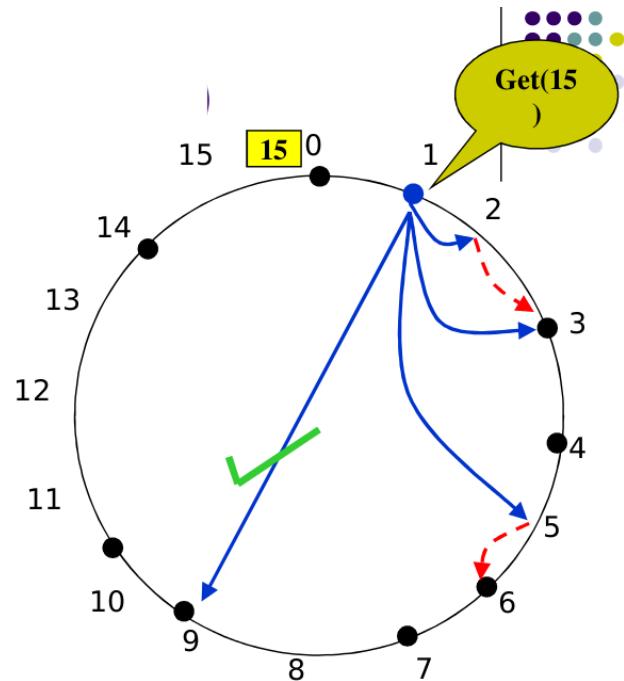
9 runs Stabilize(), i.e
First, 9 asks 12 who is your pred?
Second, 9 is notifying 12 that it
thinks it is the pred of 12



- 12 changes its predecessor to 9
- 7 will learn that 9 is its successor when it runs stabilize
- 12 will tell

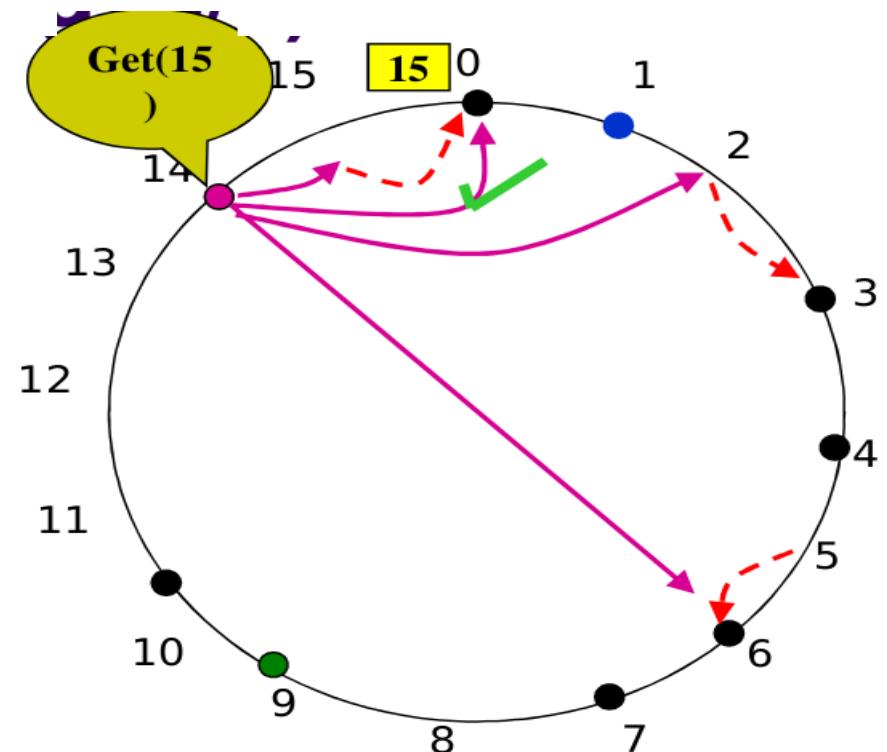
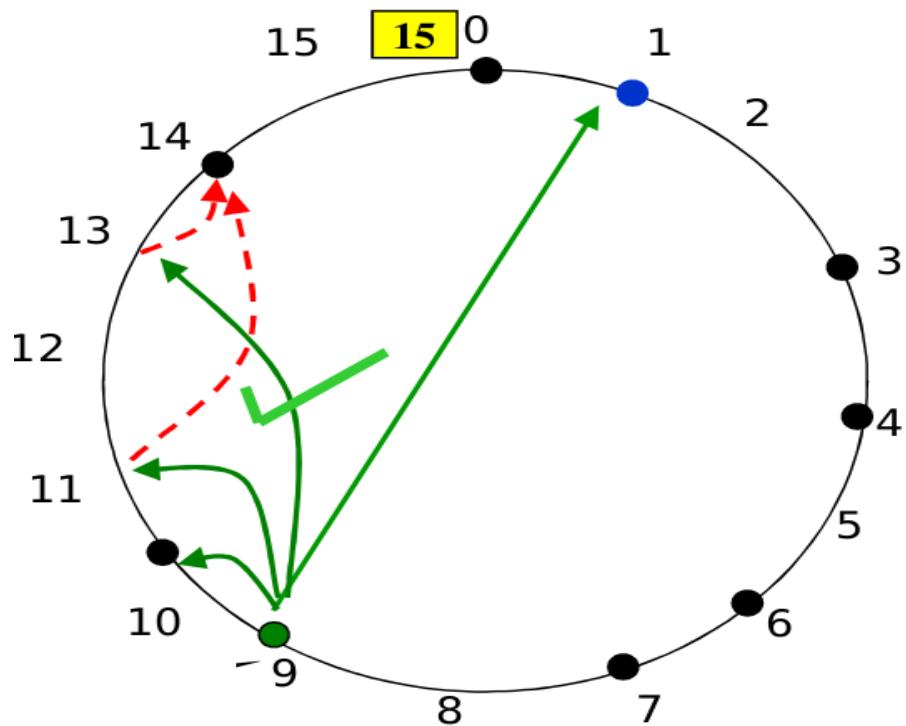
Still slow to retrieve a resource!

Routing: Fingers table



- Keep a routing table of M peers at each node
 - Where $N=2^M$
 - Maximum distance to any other node is then $\log_2(N)$, i.e., M hops

Routing: Fingers table



Perform a recursive search for the resource!

Chord: Routing

- In fact, if nodes are uniformly distributed, the maximum is **log (# of nodes)**, i.e. log (8) hops between any two nodes
- The average complexity is: **1/2 log(#nodes)**

If one successor fails,
total ring collapses?

Would not it be nice to have
something with locality?

Security of P2P networks: Sybil attacks

- Adversary introduces a large number of peers so to control the P2P system
 - Subverting the reputation system
 - Corrupting the network
 - Tricking the users
 - Controlling/influencing the network
- Problem for consensus algorithms! (Blockchain, eg Bitcoin)

Solution:
proof of work vs. proof of stake!

“One can have, some claim, as many electronic personas as one has time and energy to create.”

Judith S. Donath (pioneer of sociable media / online identity research)

Conclusion: Drawbacks of P2P systems?

- Major drawbacks with P2P systems
 - Churn and performance guarantees
 - The power of **cloud computing**
 - Spotify abandoned their P2P streaming option (2014)
 - Skype abandoned their P2P architecture (2017)
 - Decreasing cost of content delivery / more legal download options

Modern P2P networks

1. Internal within companies and datacenters (eg Dynamo at Amazon)
2. Blockchains