CSx55: Distributed Systems [P2P Systems]

Unstructured P2P Systems: Looking for something?
The traffic en route to a surge
The search unlikely to converge

You may choose to
flood peers
spawn walkers
search neighborhoods
or rely on likelihoods

The system's maintenance free
The search? Anything but

Shrideep Pallickara
Computer Science
Colorado State University

1

Frequently asked questions from the previous class survey

2

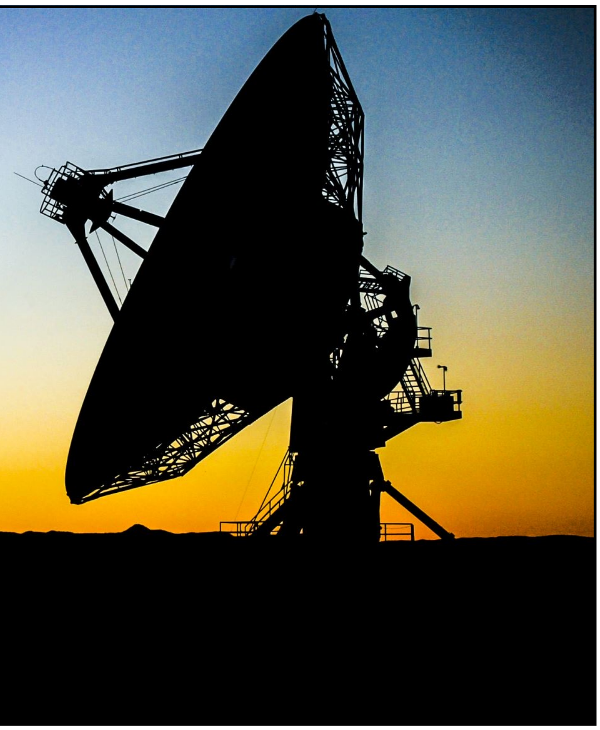
Topics covered in today's lecture

- Multicast
- Unstructured P2P Systems
- Gnutella
- BitTorrent

3

MULTICAST COMMUNICATIONS

The idea of Twitter started with me working in dispatch since I was 15 years old, where taxi cabs or firetrucks would broadcast where they were and what they were doing.
Jack Dorsey



4

Multicast communications

- Uses datagram packets
- Sender sends packet to multicast address
  - `224.0.0.0` to `239.255.255.255` **Class D**
- IPv6
  - Multicast addresses have the prefix ff00::/8

Bits	8	4	4	112
Field	Prefix	Flags	Scape	Group ID

Multicast communications

- A multicast group is specified using two elements
  - A Class D IP address
  - A standard UDP port number
- The address 224.0.0.0 is **reserved** and should not be used
- Join a multicast group by
  - First creating a `MulticastSocket` with the desired port
  - Then invoking the `joinGroup(InetAddress groupAddr)` method

IP Multicast: receiving

```
public class MulticastReceiver extends Thread {
    protected MulticastSocket socket = null;
    protected byte[] buf = new byte[256];

    public void run() {
        socket = new MulticastSocket(4446);
        InetAddress group = InetAddress.getByName("230.0.0.0");
        socket.joinGroup(group);
        while (true) {
            DatagramPacket packet = new DatagramPacket(buf, buf.length);
            socket.receive(packet);
            String received = new String(
              packet.getData(), 0, packet.getLength());
            if ("end".equals(received)) {
                break;
            }
        }
        socket.leaveGroup(group);
        socket.close();
    }
}
```

Port number → `new MulticastSocket(4446)`

Class D address → `"230.0.0.0"`

7

IP multicast: sending

8

```
public class MulticastPublisher {
    private DatagramSocket socket;
    private InetAddress group;
    private byte[] buf;

    public void multicast(
      String multicastMessage) throws IOException {
        socket = new DatagramSocket();
        group = InetAddress.getByName("230.0.0.0");
        buf = multicastMessage.getBytes();

        DatagramPacket packet
          = new DatagramPacket(buf, buf.length, group, 4446);
        socket.send(packet);
        socket.close();
    }
}
```

Class D address → `"230.0.0.0"`

Port number → `4446`

8

Multicast communications: Routing data

- Routers make sure packet delivered to all hosts in multicast group
  - Choose points where streams are *duplicated*
- Pay attention to **TTL** for the datagrams
  - Maximum number of routers a datagram is allowed to cross

Multicast issues

- **Packet size** restrictions in Multicast
- Practical considerations
  - Turned **off** at several institutions to curb *free-riding*

Multicast is not particularly suitable in some situations

- Consumption patterns change dynamically
  - Groups cannot be pre-allocated
- Representing *consumption profiles* as groups
  - Enormous number of groups – potentially 2^N for N consumers
  - Eliminating impossible groups would still require millions of groups

COLORADO STATE UNIVERSITY | Professor: SHRIDEEP PALLICKARA, COMPUTER SCIENCE DEPARTMENT | PEER-TO-PEER SYSTEMS | L18.11

11

12

Structured P2P systems [Summary]

- Overall **global policy** governing
  - Topology of the network
  - Placements of objects
  - Routing functions to locate objects
- There is a specific **distributed data structure** that underpins
  - Associated overlay
  - Algorithms that operate on it to route messages

COLORADO STATE UNIVERSITY Professor: SHRIDEEP PALLICKARA COMPUTER SCIENCE DEPARTMENT PEER-TO-PEER SYSTEMS L18.13

13

Structured P2P systems [Summary]

- Because of the structure, algorithms are
  - Efficient
  - Offer *time-bounds* on object location
- BUT involve **costly maintenance** of underlying structures
  - In highly dynamic environments

COLORADO STATE UNIVERSITY Professor: SHRIDEEP PALLICKARA COMPUTER SCIENCE DEPARTMENT PEER-TO-PEER SYSTEMS L18.14

14

Unstructured P2P systems [1/2]

- Target the maintenance argument
- No overall control on
  - Topology
  - Placements of objects within the network
- Overlay is created in an ***ad hoc*** manner
  - Each node joins network by following simple, local rules to establish connectivity

Unstructured P2P systems [2/2]

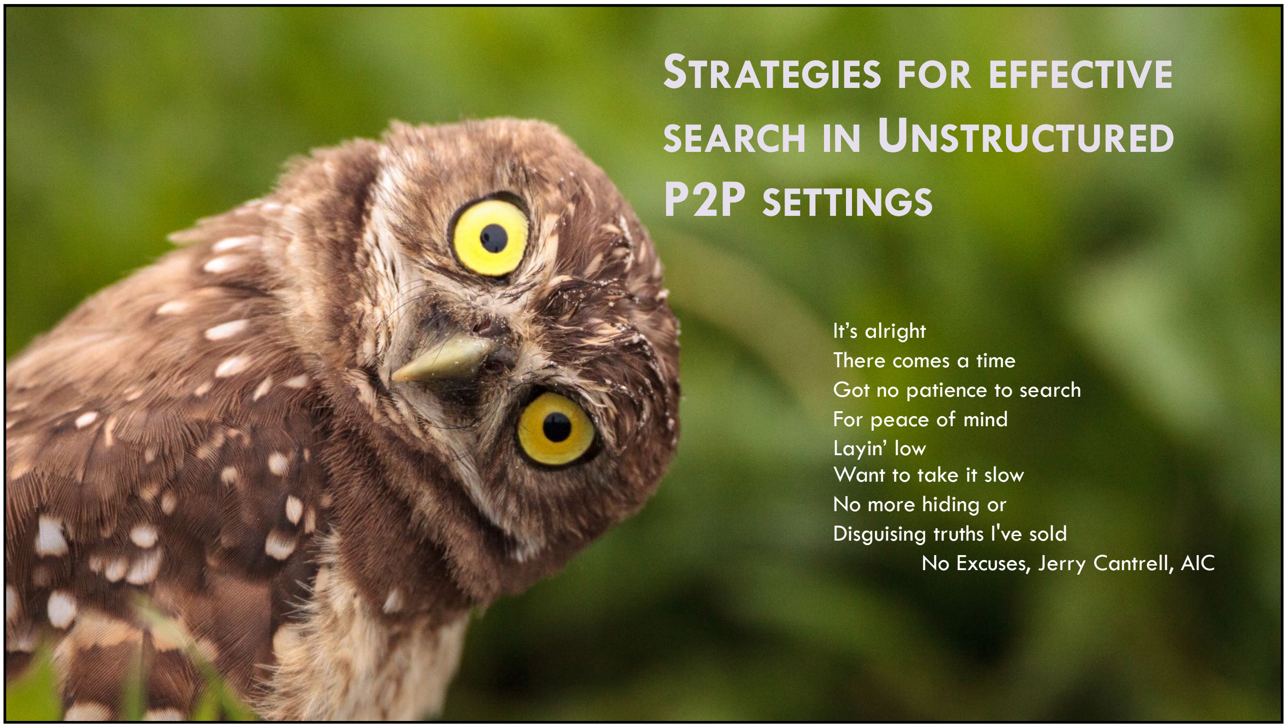
- A new joining node will establish contact with a set of *neighbor* nodes
  - These neighbors will be connected to further neighbors, etc.
- The network is fundamentally **decentralized** and self-organizing
  - Resilient to failures

Locating objects in unstructured P2P systems

- Requires a search of the resultant network topology
- **No guarantees** of being able to find the object
  - Performance will also be unpredictable
  - There is a risk of generating *excessive message traffic* to locate objects

Pros and Cons

	Structured P2P	Unstructured P2P systems
Advantages	Guaranteed to locate objects with bounds on this operation Low message overhead	Self-organizing and naturally resilient to failures
Disadvantages	Maintain complex overlay structures that are difficult and costly in dynamic settings	Probabilistic Cannot offer absolute guarantees on locating objects



19

Sharing in unstructured P2P networks

- All nodes in the network offer files to the greater environment
- Problem of locating a file?
  - Maps onto a *search of the whole network*
- CAVEAT:
  - If implemented naively, could result in **flooding** the network with requests

COLORADO STATE UNIVERSITY Professor: SHRIDEEP PALLICKARA COMPUTER SCIENCE DEPARTMENT PEER-TO-PEER SYSTEMS L18.20

20

Refinements for search in unstructured P2P systems

- Expanded ring search
- Random walks
- Gossiping
- Replication

21

Refinements for search in unstructured P2P systems: Expanded Ring Search

- Initiating node carries out a series of searches with *increasing values* in the TTL field
- A significant number of searches will likely be satisfied locally (proximate peers)
  - Expand the scope of search only if requests fail in the neighborhood

22

Refinements for search in unstructured P2P systems: Random Walks

- Initiating node sets of a number of **walkers**
- Walkers follow *random pathways* through the interconnected graph
  - Over the unstructured network

Refinements for search in unstructured P2P systems: Gossiping [1/2]

- Node sends request to a neighbor with a certain probability
- Requests propagate through the network in a manner that is similar to **viral propagations**
  - Such gossip protocols are also referred to as *epidemic protocols*

Refinements for search in unstructured P2P systems: Gossiping [2/2]

- Probabilities may either be
  - **Fixed** for a given network
  - Computed **dynamically** based on:
    - Past experience
    - Current context

COLORADO STATE UNIVERSITY | Professor: Shrideep Pallickara, Computer Science Department | Peer-to-Peer Systems | L18.25

25

Refinements for search in unstructured P2P systems: Replication

- **Replicate** content across a number of peers
- *Probability* of efficient discovery for particular files is enhanced
- Replications can be for
  - The entire file
  - Fragments thereof

COLORADO STATE UNIVERSITY | Professor: Shrideep Pallickara, Computer Science Department | Peer-to-Peer Systems | L18.26

26

GNUTELLA

COLORADO STATE UNIVERSITY

27

Gnutella

- Launched in 2000
- One of the most dominant and influential peer-to-peer file sharing applications

COLORADO STATE UNIVERSITY   Professor: SHRIDEEP PALLICKARA COMPUTER SCIENCE DEPARTMENT   PEER-TO-PEER SYSTEMS   L18.28

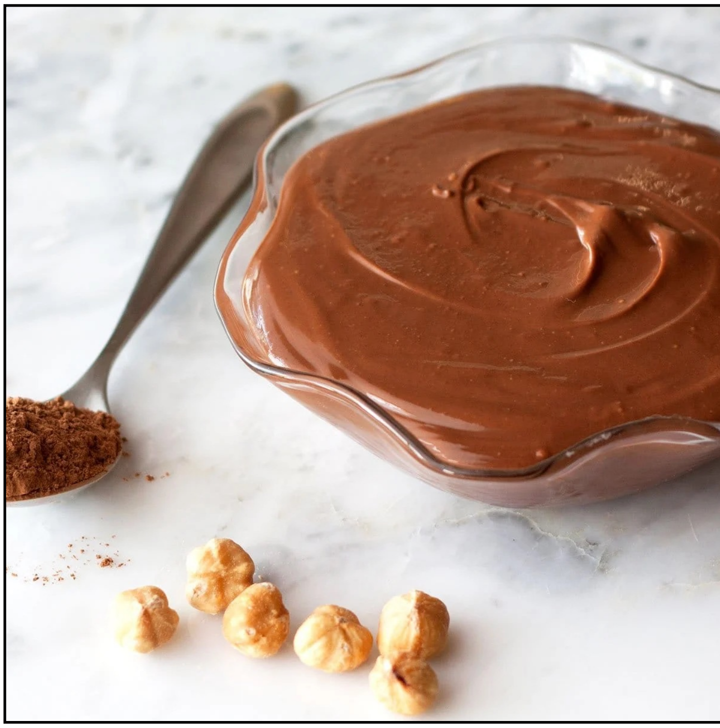
28

Gnutella: Early Versions (0.4)

- Every node forwarded a request to **each** of its neighbors
- Neighbors, in turn, passed this on to their neighbors
  - Until a match was found
- This is **flooding**

Gnutella: Early Versions (0.4)

- Search was **constrained** with a *time-to-live* (TTL) field limiting the number of hops
- At the time of Version 0.4, average peer connectivity was 5 neighbors per-node



GNUTELLA
VERSION 0.6 AND LATER

COLORADO STATE UNIVERSITY

31

Addressing deficiencies in scaling: Hybrid Architecture [1/2]

- Move away from classic P2P where all nodes are equal
- Some nodes are elected as **ultrapeers**
  - Form the heart of the network
- Other nodes take on the role of **leaf nodes**
- Peers still cooperate to offer service

32

Addressing deficiencies in scaling: Hybrid Architecture [2/2]

- Leaves connect to a small number of **ultrapeers**
- Ultrapeers are *densely connected* to other ultrapeers
- Effect?
  - Dramatically *reduces* the maximum number of hops for exhaustive search

COLORADO STATE UNIVERSITY Professor: SHRIDEEP PALLICKARA COMPUTER SCIENCE DEPARTMENT PEER-TO-PEER SYSTEMS L18.33

33

Query Routing Protocol [1/2]

- Designed to **reduce** the number of queries issued by nodes
- **Exchange information** about files contained on nodes
- **Forward queries** down paths where the system thinks there will be a positive outcome

COLORADO STATE UNIVERSITY Professor: SHRIDEEP PALLICKARA COMPUTER SCIENCE DEPARTMENT PEER-TO-PEER SYSTEMS L18.34

34

Query Routing Protocol [2/2]

- Does not share information about files directly
- Protocol produces **set of numbers**
  - By *hashing on individual words* in a file-name
  - For e.g., "`Gone with the wind`" will be represented as <`36, 789, 452, 132`>
- Each node produces a **Query Routing Table**
  - Contains hash values representing *each of the files* contained on that node
  - Sends it to all its associated ultrapeers

Query Routing Protocol: Ultrapeers

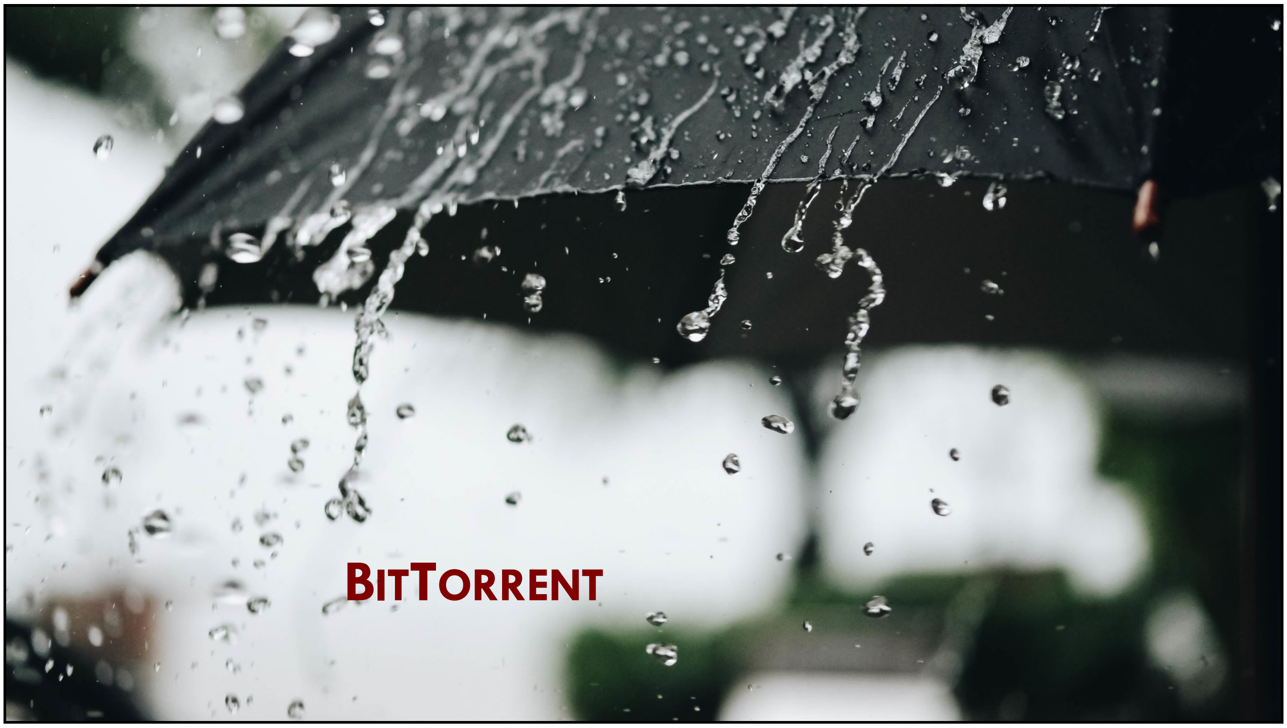
- Ultrapeers produce their own Query Routing Table
  - **Union** of all entries *from all connected leaves*; together with entries for files at that ultrapeer
- The ultrapeer then **exchanges** its Query Routing Table with other ultrapeers

Implications of exchanging the Query Routing Table

- Ultrapeers can determine which paths offer a **valid route** for a given query
  - Significantly reduces amount of unnecessary traffic
- Ultrapeer **forwards** a query to a node *only if a match is found*
  - Match indicates that the node has the file
  - Same check performed before forwarding query to another ultrapeer

Avoid overloading the ultrapeers

- Nodes send query to *one* ultrapeer at a time
  - Wait for a specified time period
- ***Avoid reverse traversal*** of messages through the graph
  - Queries in Gnutella contain network address of the initiating ultrapeer
  - File sent directly (using UDP) to that ultrapeer



39

Bit Torrent: Traffic statistics

- In November 2004
  - Responsible for 25% of all Internet traffic
- February 2013
  - 3.35% of all worldwide bandwidth
  - > 50% of the 6% total bandwidth dedicated to file sharing

40

Other places where BitTorrent is used

- Facebook
  - To distribute updates to Facebook servers
- Twitter
  - To distribute updates to Twitter servers
- The British government
  - Used BitTorrent to distribute details about how the tax money of British citizens was spent

BitTorrent

- Designed for downloading **large files**
- Not intended for real-time routing of content
- Relies on capabilities of ordinary user machines

Bit Torrent: Key concepts

- Instead of downloading a file from a single source server
  - Users join a **swarm** of hosts to upload-to/download-from *simultaneously*
- Several basic commodity computers can replace large, customized servers
  - Without compromising on efficiency
  - In fact, lower bandwidth usage with swarms *prevents* large internet traffic spikes

Segmented file transfer [1/2]

- File being transferred is divided into fixed-size segments called **chunks** (or pieces)
  - Chunks are of the same size throughout a single download (10MB file: 10 1MB chunks or 40 256KB chunks)
- Chunks are downloaded non-sequentially and rearranged into the correct order by BitTorrent

Segmented file transfer [2/2]

- Advantages:
  - File transfers can be stopped at any time and *resumed*
    - Without loss of previously downloaded content
  - Clients seek out readily available chunks, rather than waiting for an unavailable (next in sequence) chunk

COLORADO STATE UNIVERSITY Professor: SHRIDEEP PALLICKARA COMPUTER SCIENCE DEPARTMENT PEER-TO-PEER SYSTEMS L18.45

45

The contents of this slide-set are based on the following references

- *Distributed Systems: Principles and Paradigms. Andrew S. Tanenbaum and Maarten Van der Steen. 2nd Edition. Prentice Hall. ISBN: 0132392275/978-0132392273.* [Chapter 5]
- *Distributed Systems: Concepts and Design. George Coulouris, Jean Dollimore, Tim Kindberg, Gordon Blair. 5th Edition. Addison Wesley. ISBN: 978-0132143011.* [Chapter 10]
- *Broadcasting and Multicasting in Java: https://www.baeldung.com/java-broadcast-multicast*

COLORADO STATE UNIVERSITY Professor: SHRIDEEP PALLICKARA COMPUTER SCIENCE DEPARTMENT PEER-TO-PEER SYSTEMS L18.46

46