Peer-to-Peer Networks

南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

Xuetao Wei

weixt@sustech.edu.cn

Outline

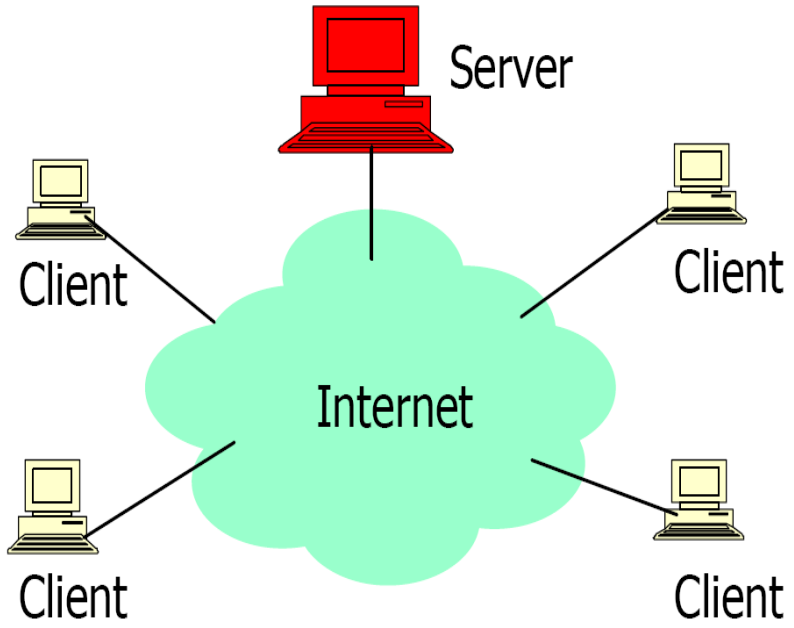
南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

- **P2P: concepts & architecture**
- Content distribution
- P2P file sharing
- Other P2P applications

Client/Server Architecture

南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

- Well known, powerful, reliable server is a data source
- Clients request data from server
- Very successful model (WWW, FTP, Web services, etc)

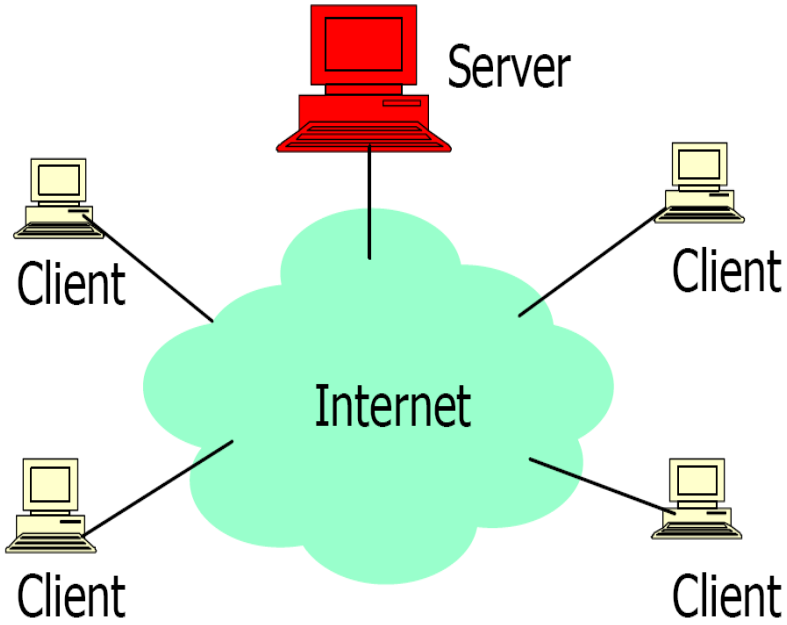


As more clients are added, demand on the server increases!

Client/Server Limitations

南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

- Scalability is hard to achieve
- Presents a single point of failure
- Requires administration
- Unused resources at the network edge
  - CPU cycles, storage, etc



P2P systems try to address these limitations...

Why Study P2P

南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

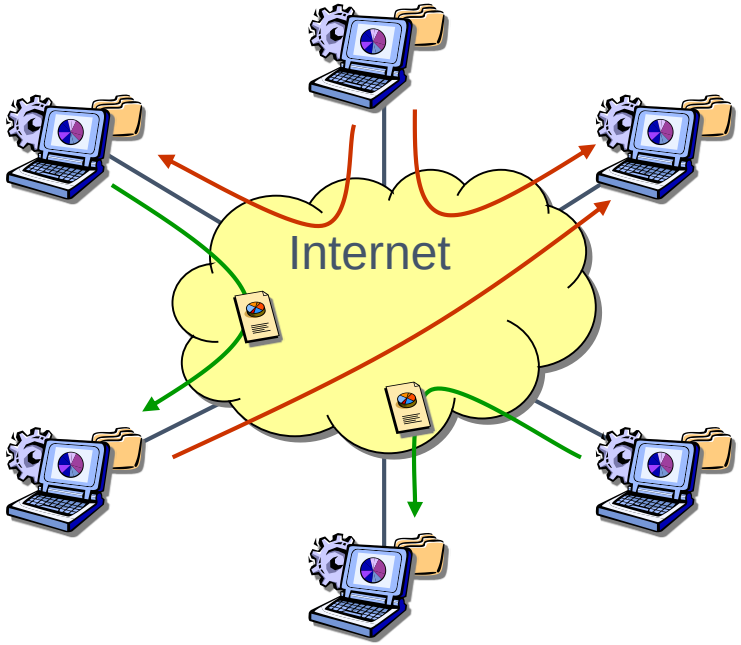
- Huge fraction of traffic on networks today: >= 50%!
- Many exciting new applications
- Next level of resource sharing

P2P Architecture

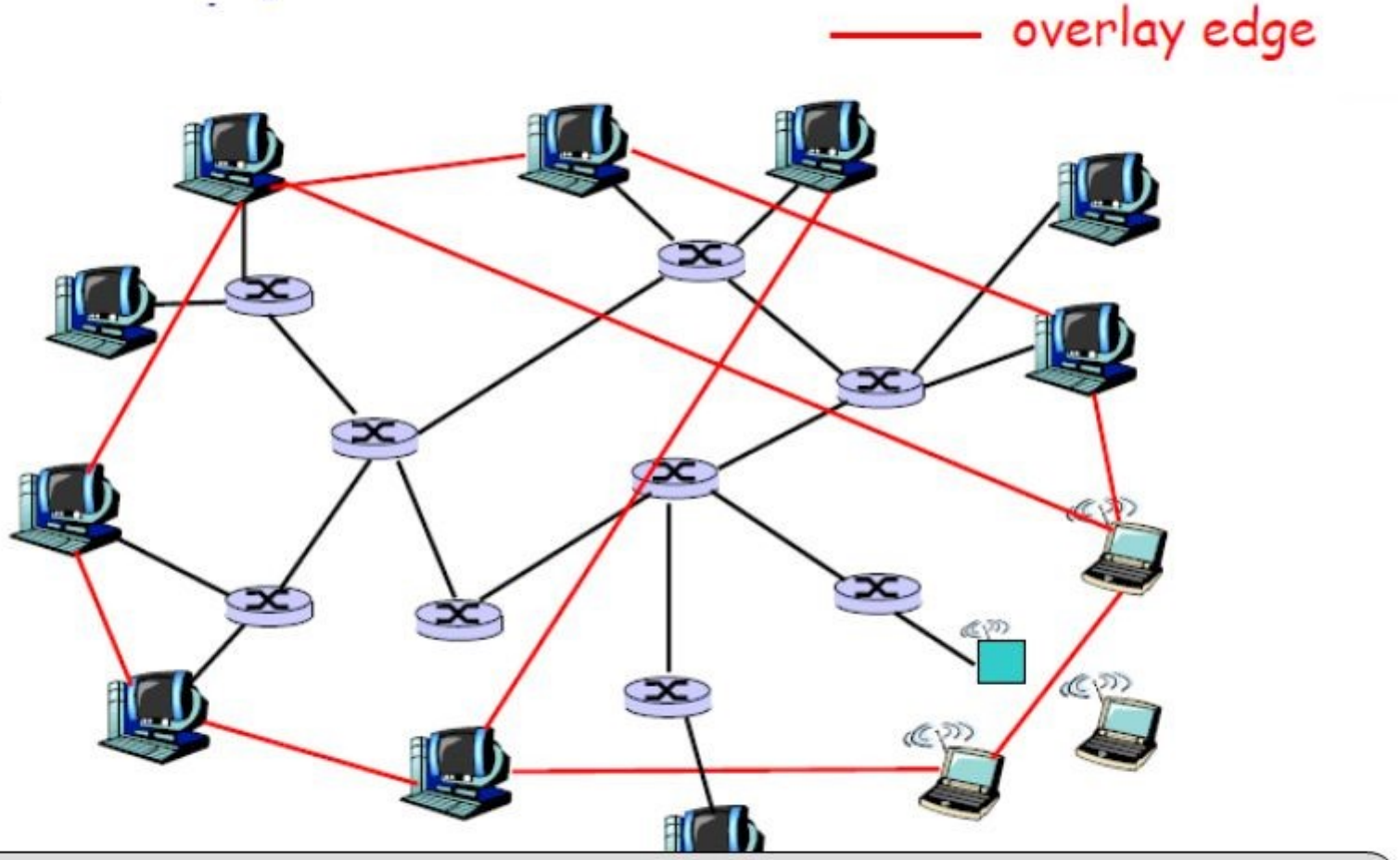
P2P Architecture最精简的描述

南方科技大学 SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

- **All nodes are both clients and servers**
  - Provide and consume data
  - Any node can initiate a connection
- **No centralized control**
  - Many nodes, heterogeneous and unreliable
  - But the system is self-organizing & fault-tolerant
- **No centralized data source**
  - "The ultimate form of democracy on the Internet"
  - "The ultimate threat to copy-right protection on the Internet"



Overlay Network

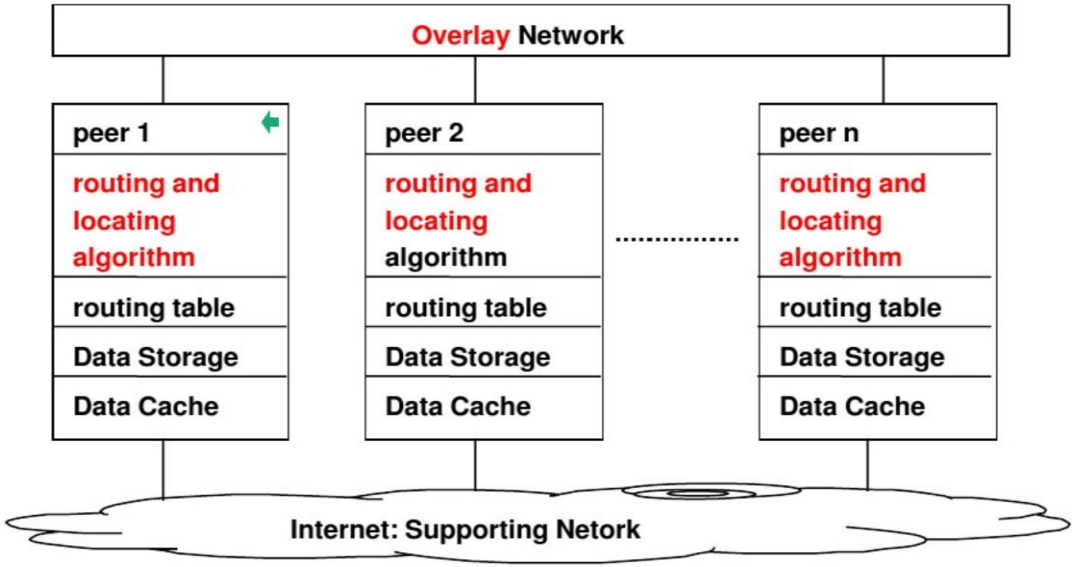


A P2P network is an overlay network. Each link between peers consists of one or more IP links.

Overlays : All in the Application Layer

南方科技大学 SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

- **Tremendous design flexibility**
  - Topology, maintenance
  - Message types
  - Protocol
  - Messaging over TCP or UDP
- **Underlying physical network is transparent to developers**
  - But some overlays exploit proximity



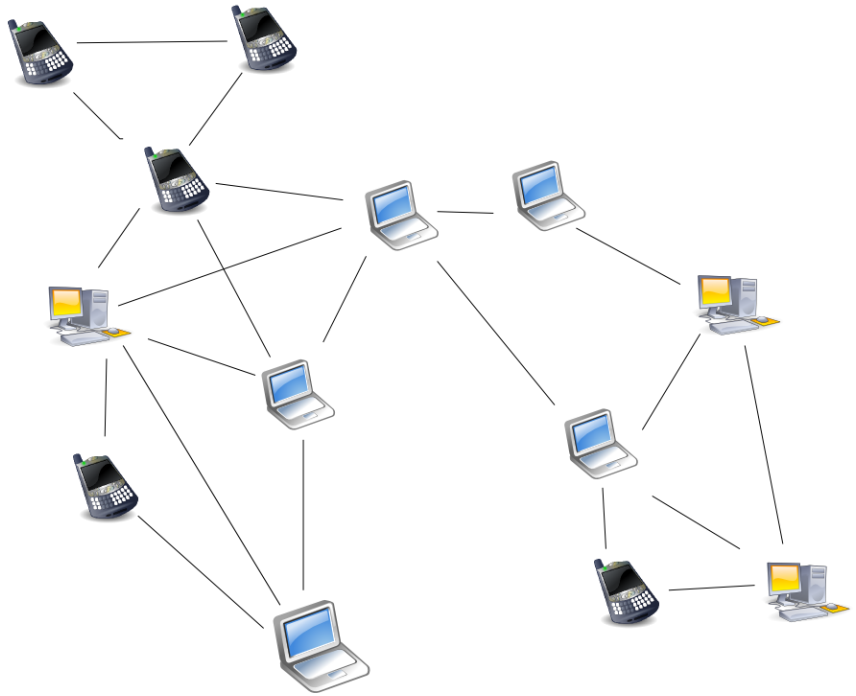
A Generic Topological Model of P2P Systems

Architectures of P2P

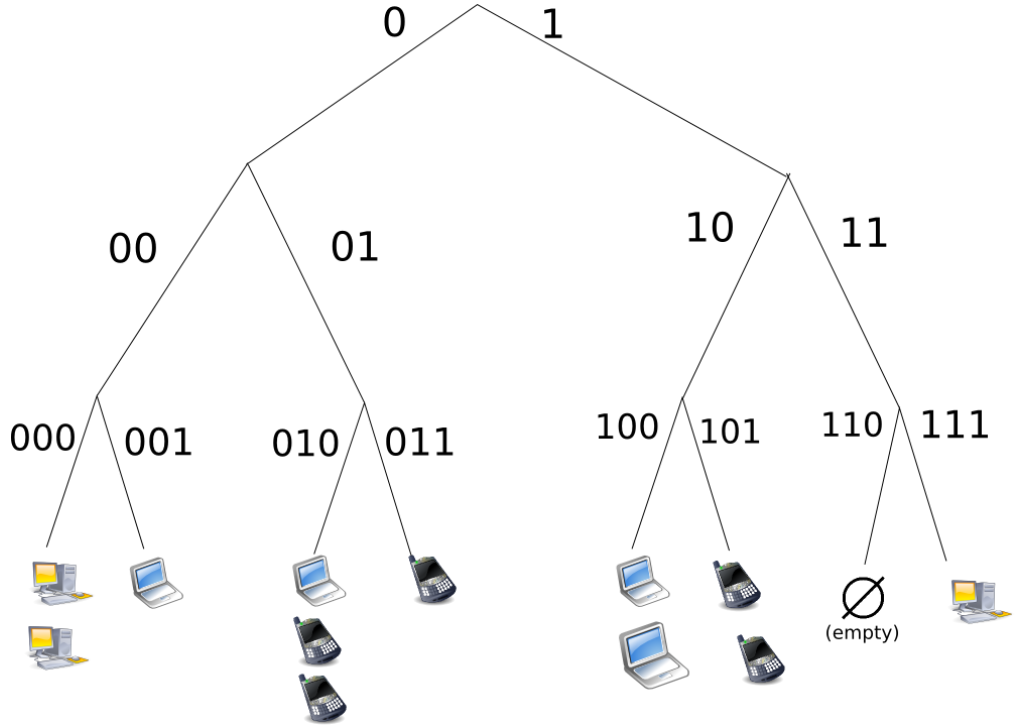
南方科技大学 SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

- **Unstructured overlays**
  - No particular structure on the overlay network, but formed by nodes that randomly connected to each other
  - Easy to build, highly robust on high rates of "churn"
  - Flooding for content search query is very inefficient

  重要
- **Structured overlays**
  - The overlay is organized into a specific topology
  - Any node can efficiently search the network for a file/resource, even if the resource is extremely rare
  - Commonly based on Distributed Hash Table (DHT)
  - Less robust in networks with a high rate of churn
  - Load imbalance problem



Unstructured Overlay Example



Structured Overlay Example

Applications

- **Content distribution**
  - Users both provide and use resources, the more users, the higher capacity for content-serving
  - Protocols: BitTorrent
- **P2P file sharing**
  - Unstructured: Napster, Gnutella, eDonkey, ...
  - Structured: Chord, CAN, Kademlia, ...
- **Multimedia**
  - Protocols: P2PTV, PDTP
- **P2P Communications**
  - Skype, PPTV, Social Networking Apps
- **Blockchain**
  - Bitcoin, ...

Key Issues

重要

- **Management**
  - How to maintain the P2P system under high rate of churn efficiently
  - Application reliability is difficult to guarantee
- **Lookup**
  - How to find out the appropriate content/resource that a user wants
- **Throughput**
  - Content distribution/dissemination applications
  - How to copy content fast, efficiently, and reliably

Management Issue

南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

- A P2P network must be self-organizing
  - Join and leave operations must be self-managed
  - The infrastructure is untrusted and the components are unreliable
  - The number of faulty nodes grows linearly with system size
  - Tolerance to failures and churn
    - Content replication, multiple paths
    - Leverage knowledge of executing application
- Load balancing
- Dealing with **freeriders**
  - Freerider : rational or selfish users who consume more than their fair share of a public resource

Lookup Issues

- How do you locate data/files/objects in a large P2P system built around a dynamic set of nodes in a scalable manner without any centralized server or hierarchy?

- Efficient routing even if the structure of the network is unpredictable
  - **Unstructured P2P** : Napster, Gnutella, Kazaa, BitTorrent
  - **Structured P2P** : Chord, CAN, Pastry/Tapestry, Kademlia

Outline

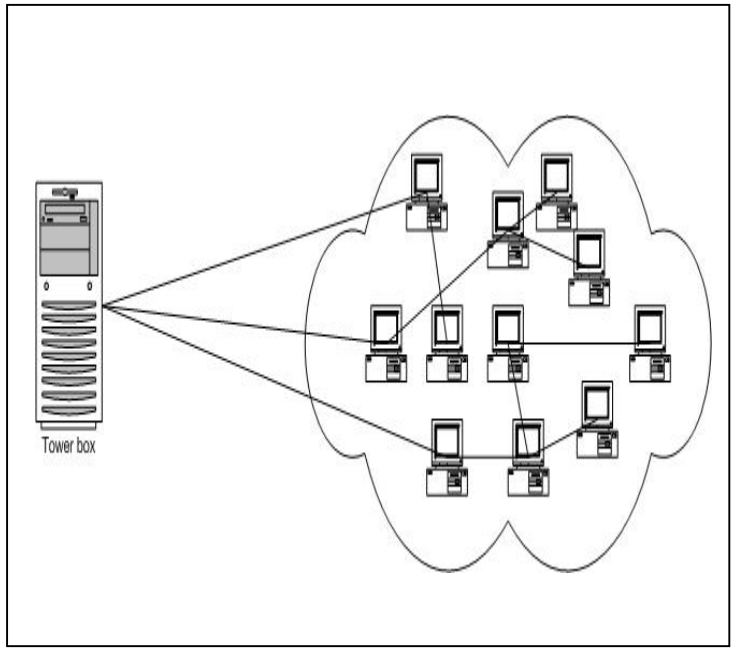
南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

- P2P: concepts & architecture
- **Content distribution**
- P2P file sharing
- Other P2P applications

Content Distribution

南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

- **Least time to disseminate:**
  - Fixed data *D* from one seeder to *N* nodes
- **Insights / Axioms**
  - Involving end-nodes speeds up the process (Peer-to-Peer)
  - Chunking the data also speeds up the process
- **Raises many questions**
  - How do nodes find other nodes for exchange of chunks?
  - Which chunks should be transferred?
  - Is there an optimal way to do this?

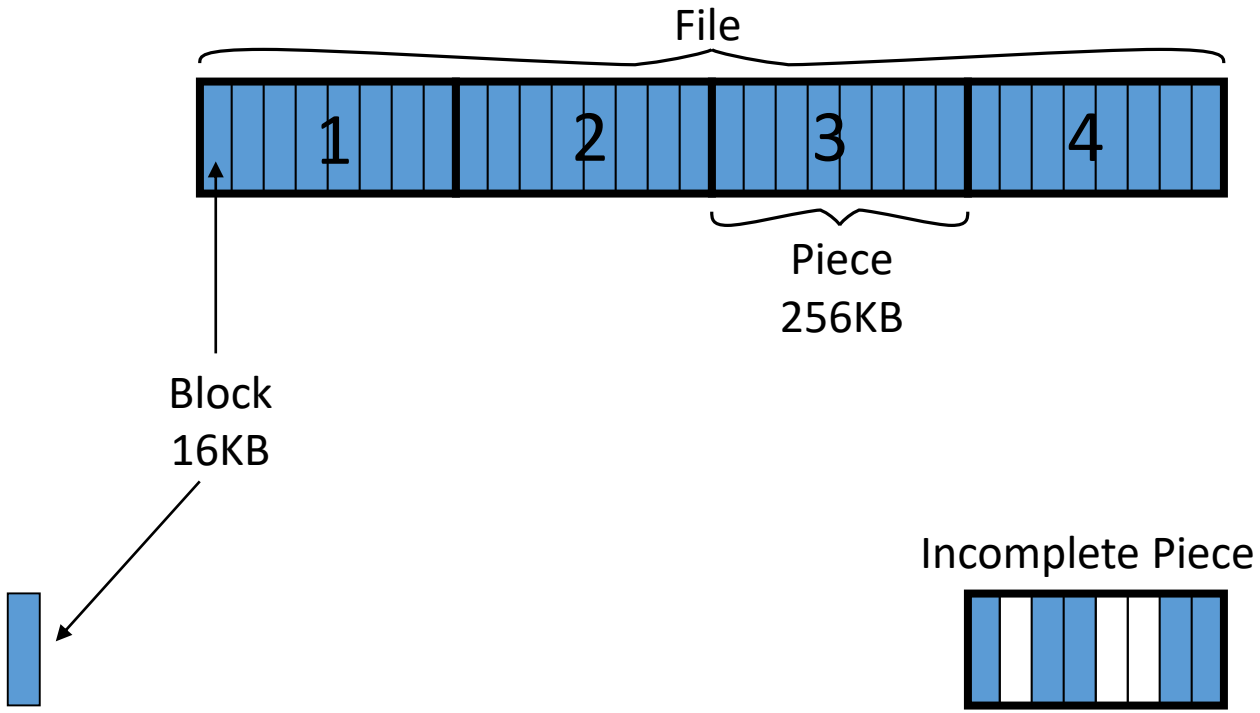


BitTorrent

南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

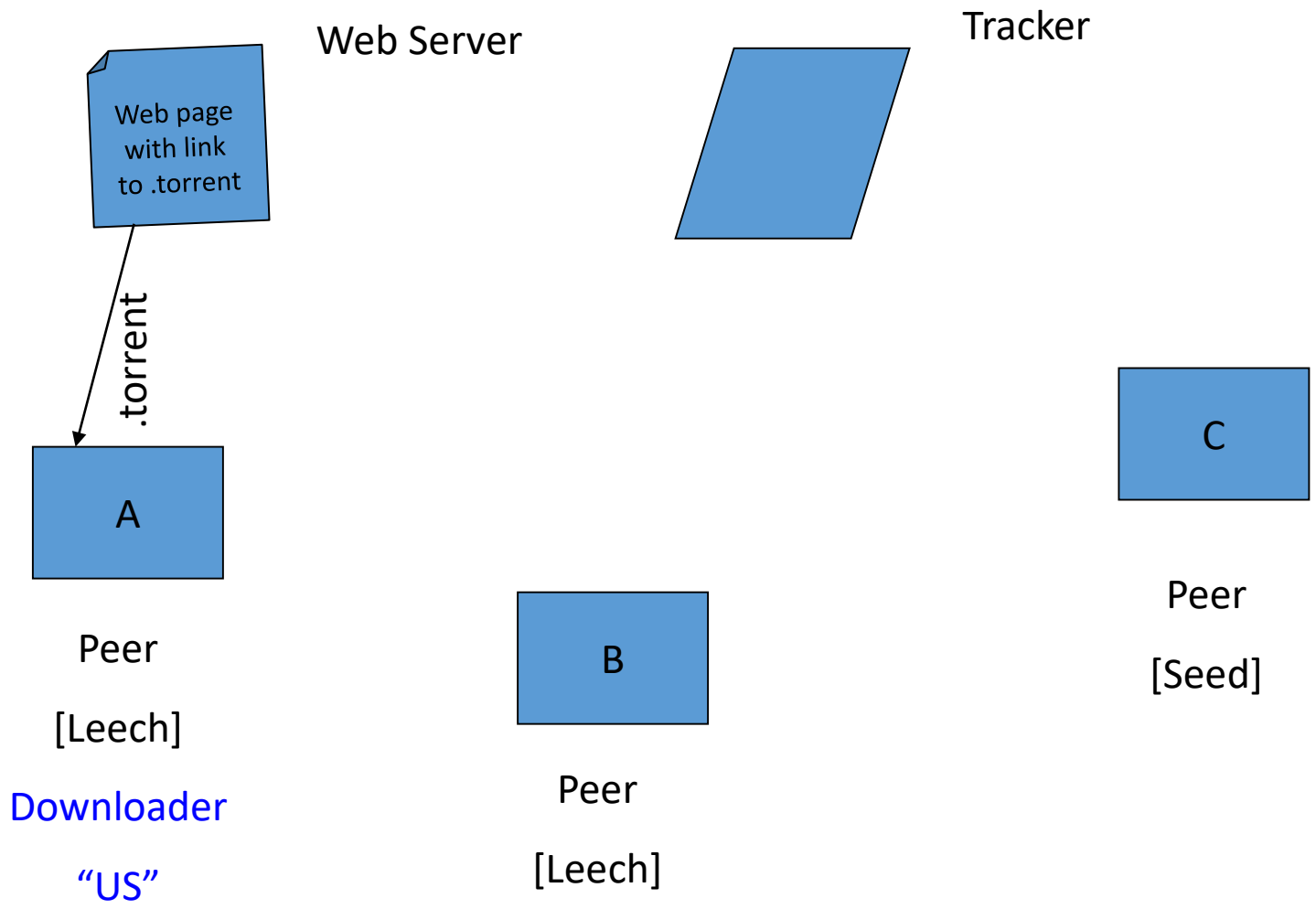
- BitTorrent was responsible for 3.35% of all worldwide bandwidth (2013)
- Special client software is needed
  - BitTorrent, BitTyrant, µTorrent, LimeWire ...
- Basic idea
  - Chop file into many pieces (chunks), as soon as a peer has a complete chunk, it can trade it with other peers
  - Clients that download a file at the same time help each other (ie, also upload chunks to each other)
  - BitTorrent clients form a swarm : a random overlay network

File Organization

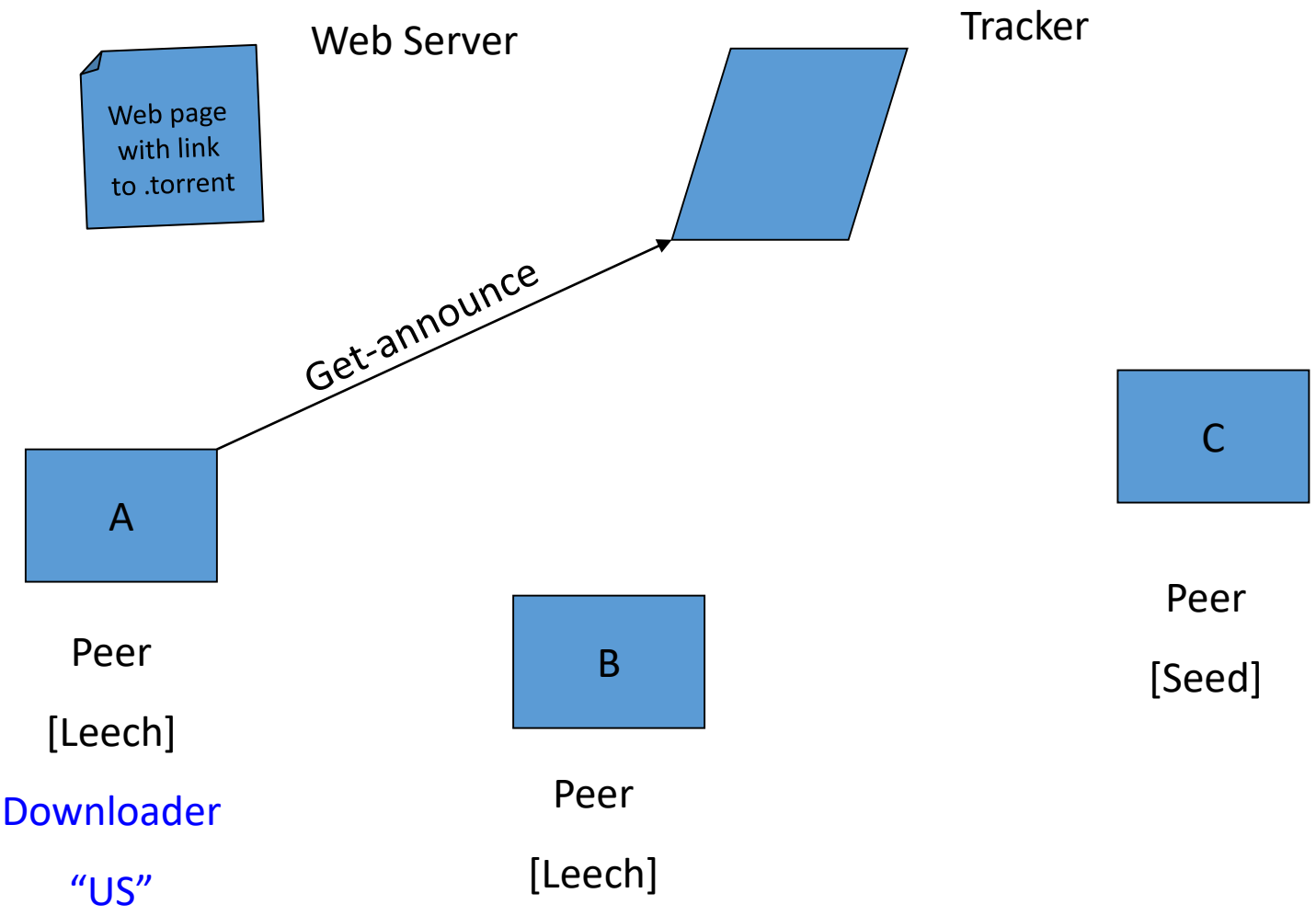


Overall Architecture

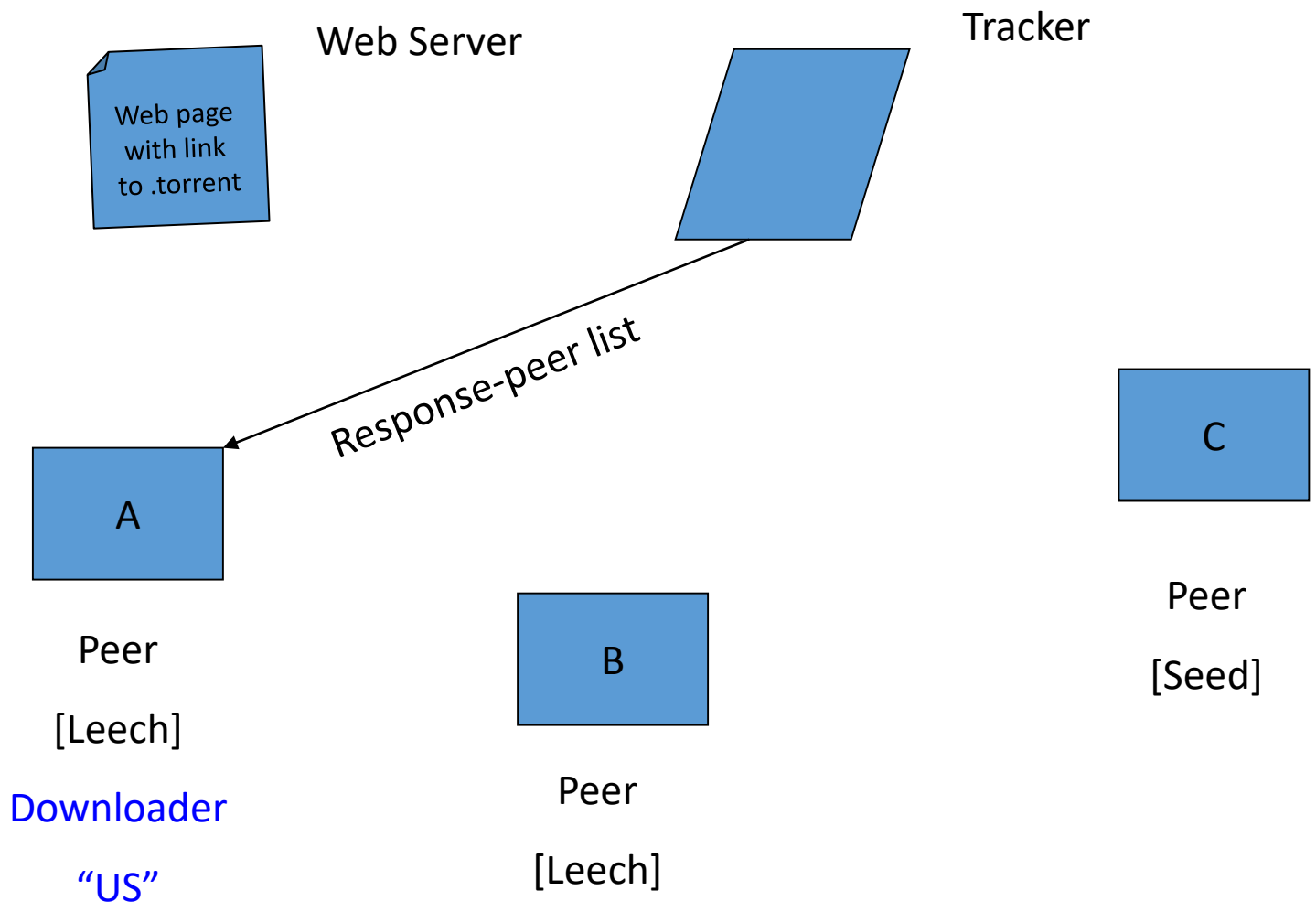
重要



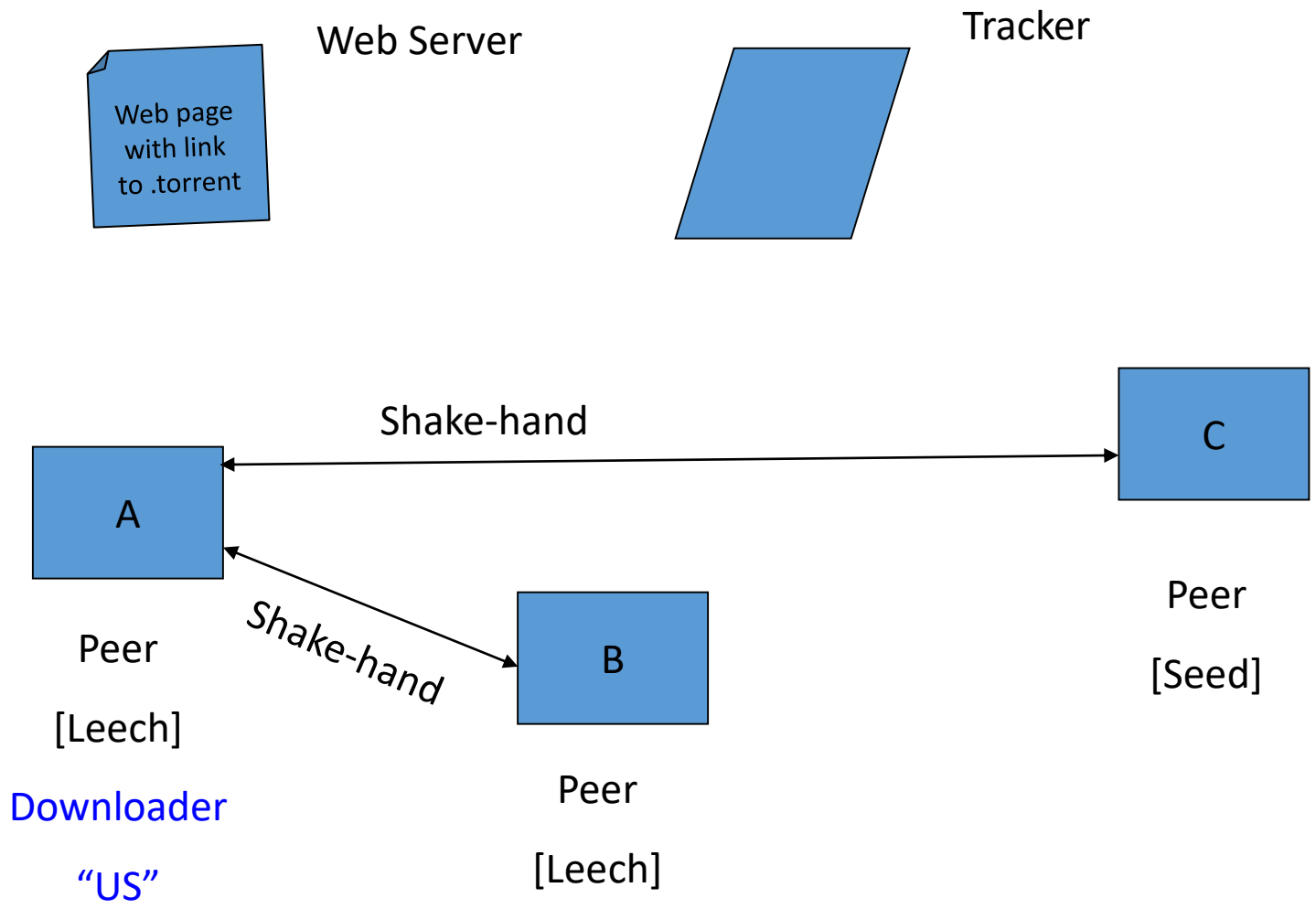
Overall Architecture



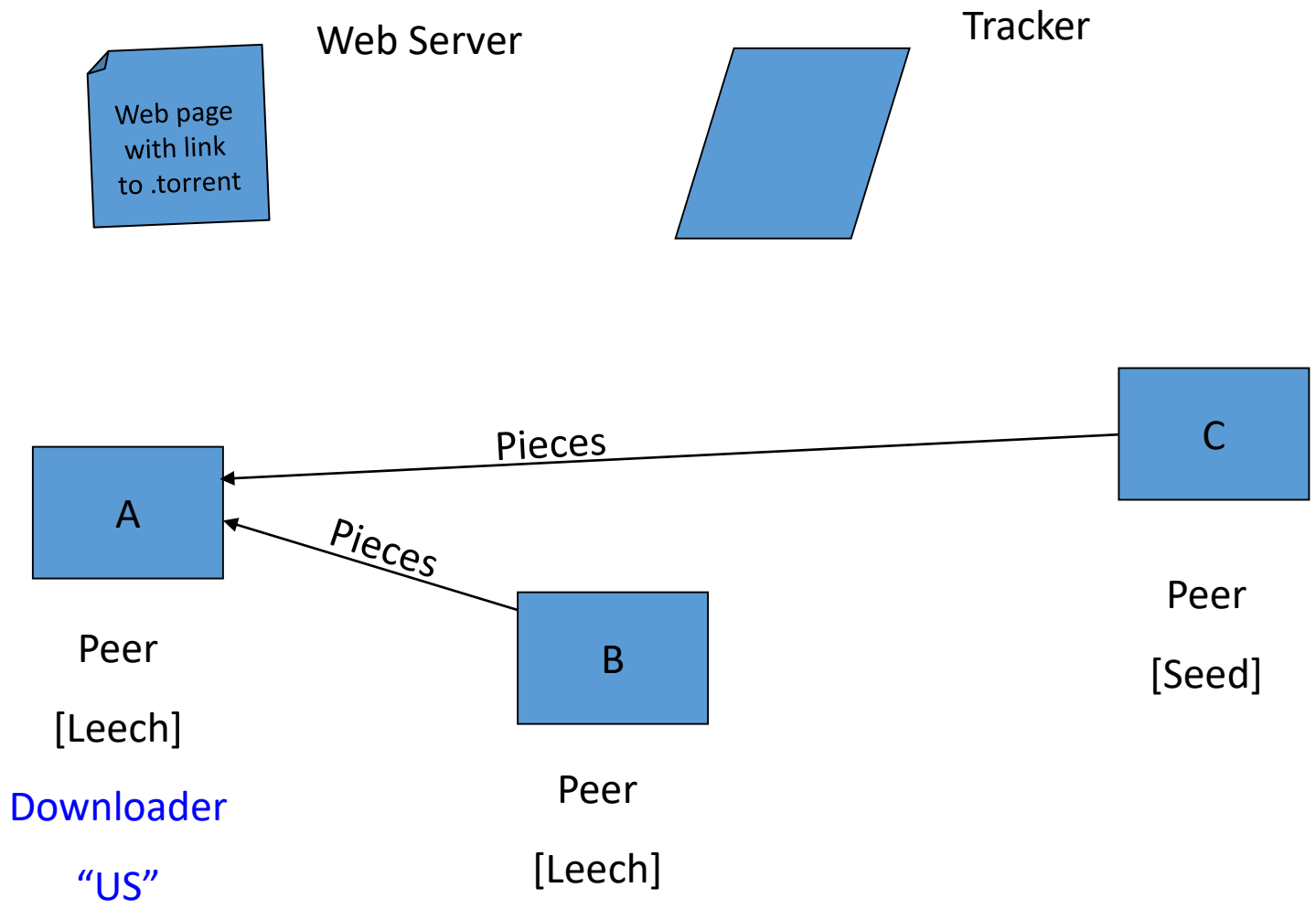
Overall Architecture



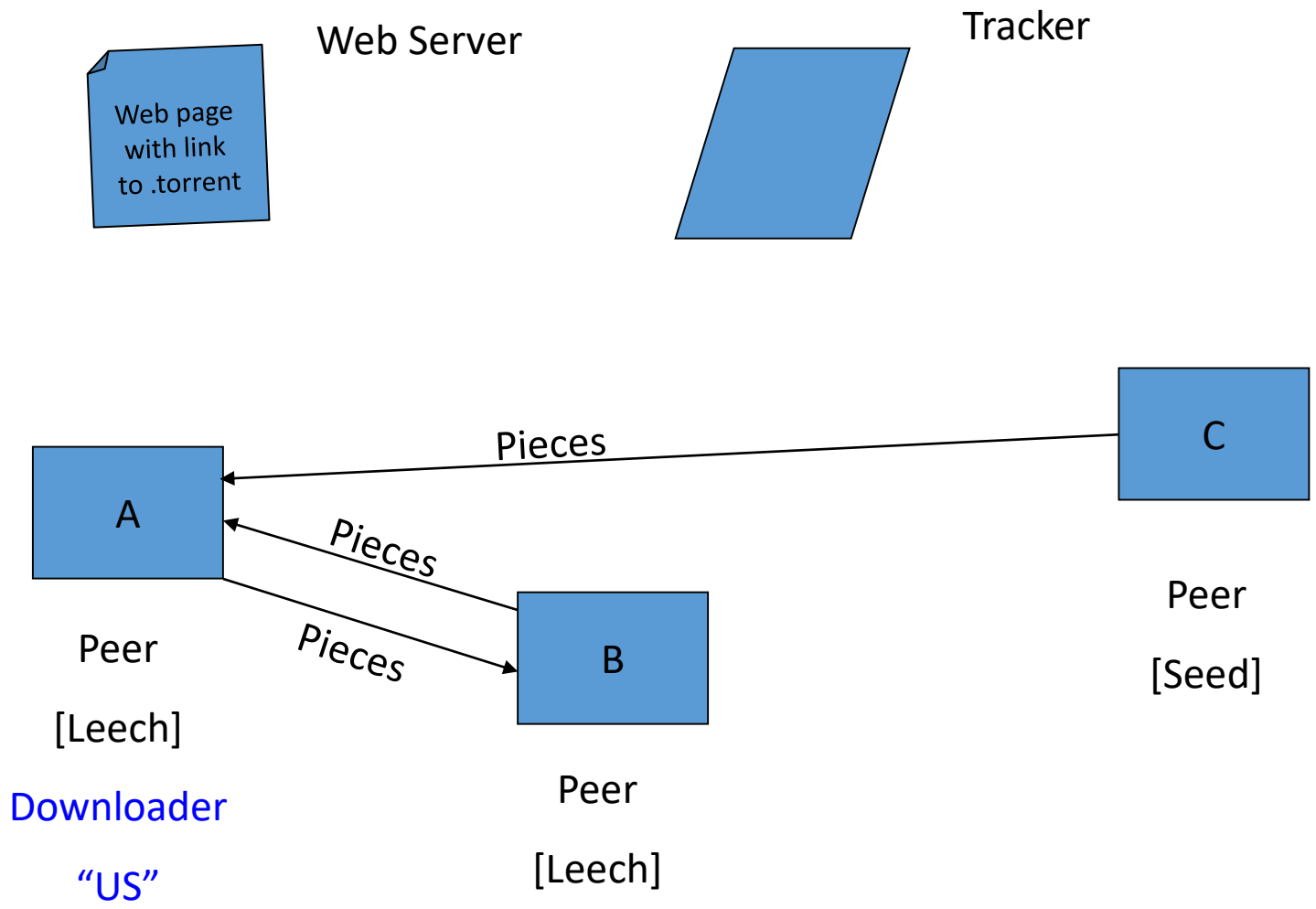
Overall Architecture



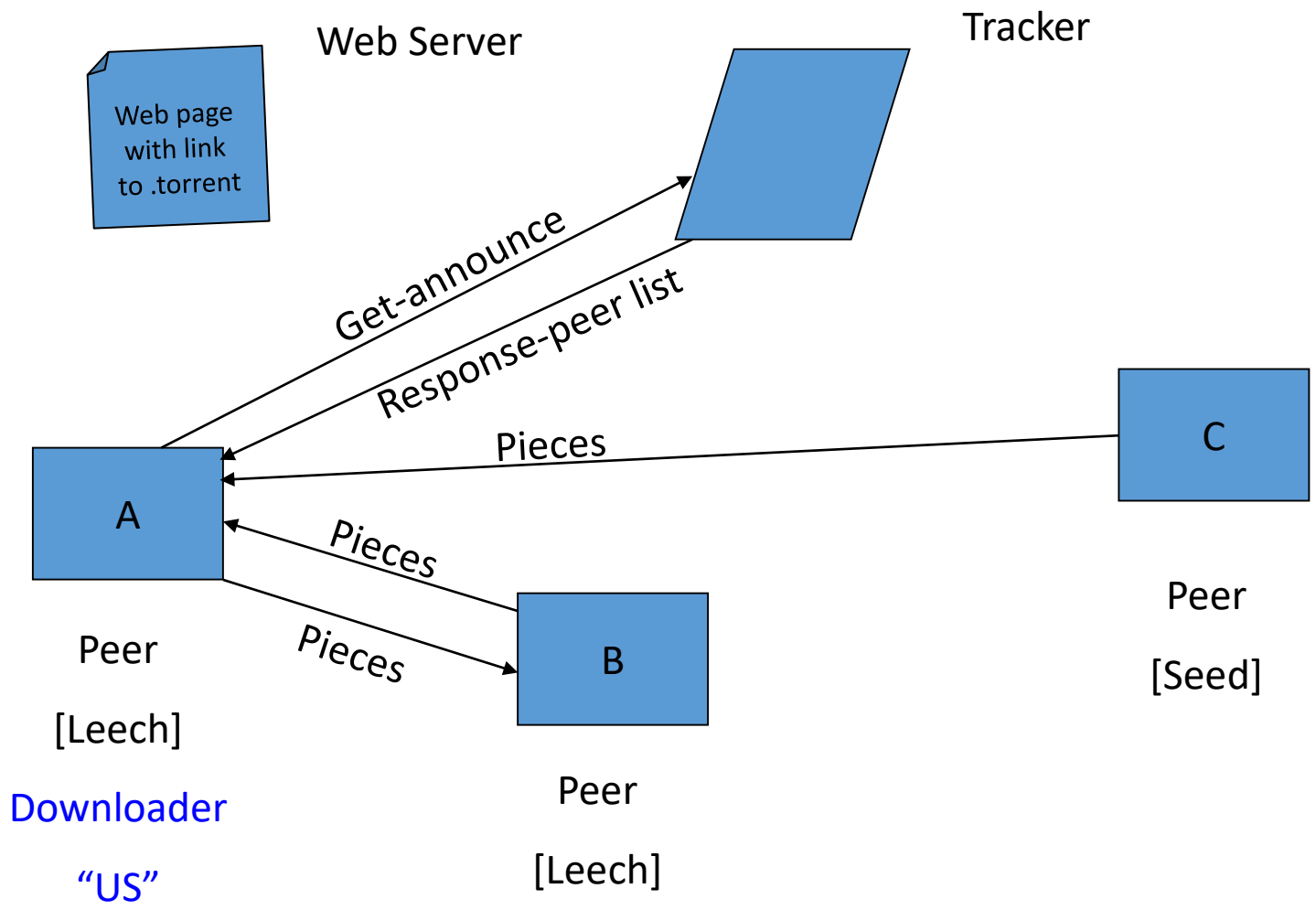
Overall Architecture



Overall Architecture



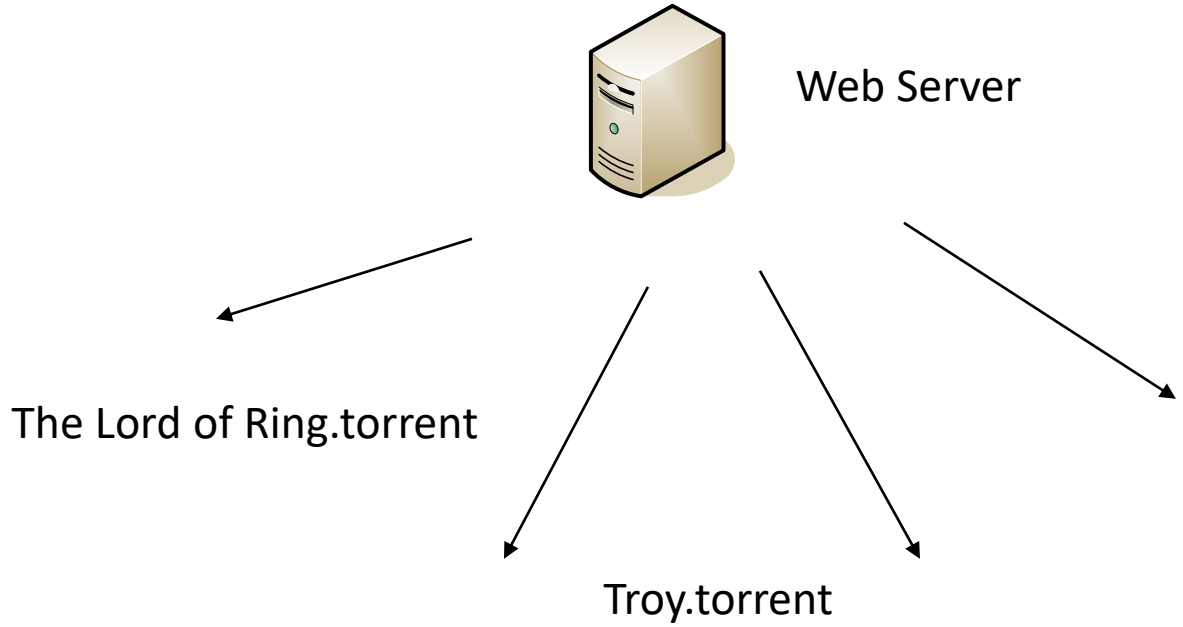
Overall Architecture



Critical Elements: A Web Server

南方科技大学 SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

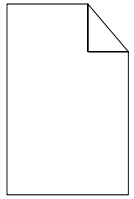
- To provide the “metainfo” file by HTTP
- For example
  - http://bt.btchina.net
  - http://bt.ydy.com



Critical Elements: .torrent File

南方科技大学 SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

- Static “metainfo” file to contain necessary information :
  - Name
  - Size
  - Checksum
  - IP address (URL) of the Tracker
  - Pieces <$hash_1$,$hash_2$,….$hash_n$>
  - Piece length



Matrix.torrent

Critical Elements: A BitTorrent Tracker

南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

- Non-content-sharing node
- Track peers
- For example:
  - http://bt.cnxp.com:8080/announce
  - http://btfans.3322.org:6969/announce

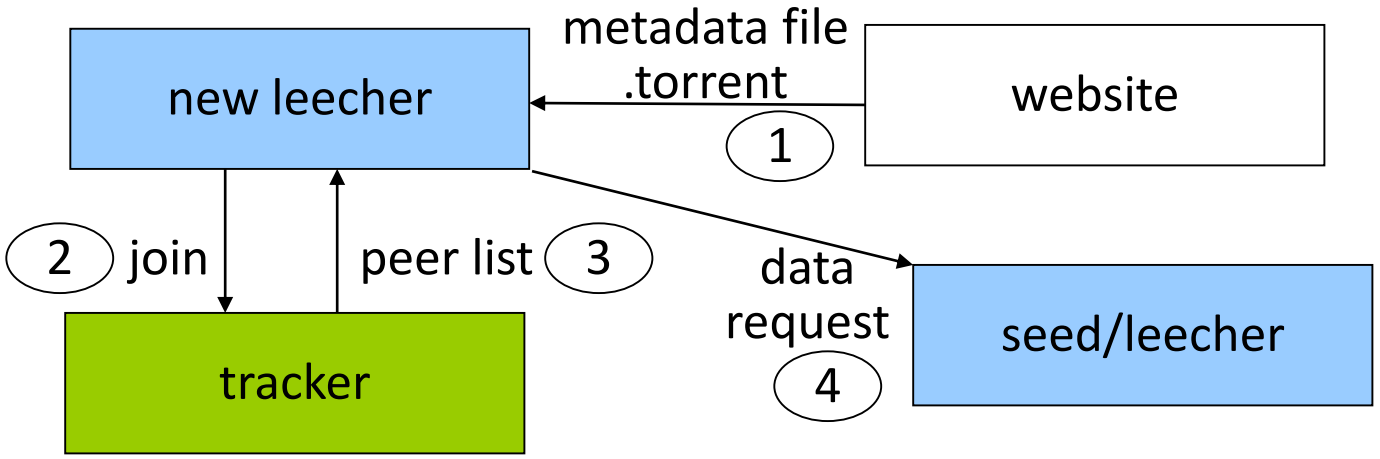
- Peer cache
  - IP, port, peer id
- State information
  - Completed
  - Downloading
- Returns random list

Critical Elements: End User (Peer)

- Guys who want to use BitTorrent must install corresponding software or plug-in for web browsers

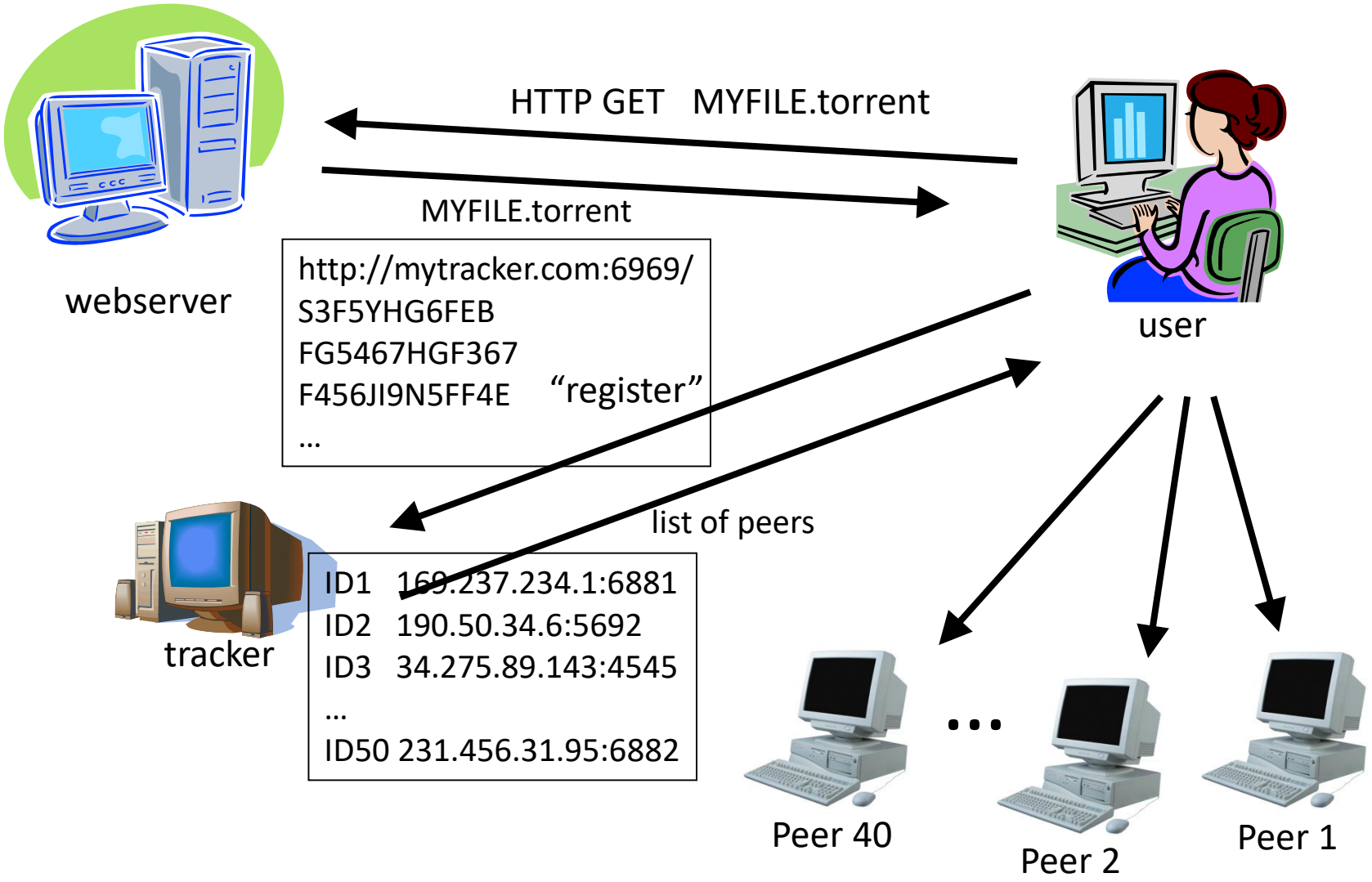
- **Leecher** (Downloader)
  - Peer has only a part (or none) of the file
- **Seeder**
  - Peer has the complete file, and chooses to stay in the system to allow other peers to download

Messages



- Peer – Peer messages
  - TCP Sockets
- Peer – Tracker messages
  - HTTP Request/Response

BitTorrent: Demo



BitTorrent : Pros/Cons

南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

- **Pros**
  - Proficient in utilizing partially downloaded files
  - Encourages diversity through “rarest-first”
    - Extends lifetime of swarm
  - Works well for “hot content”

- **Cons**
  - Assumes all interested peers active at same time; performance deteriorates if swarm “cools off”
  - Even worse: no trackers for obscure content

Outline

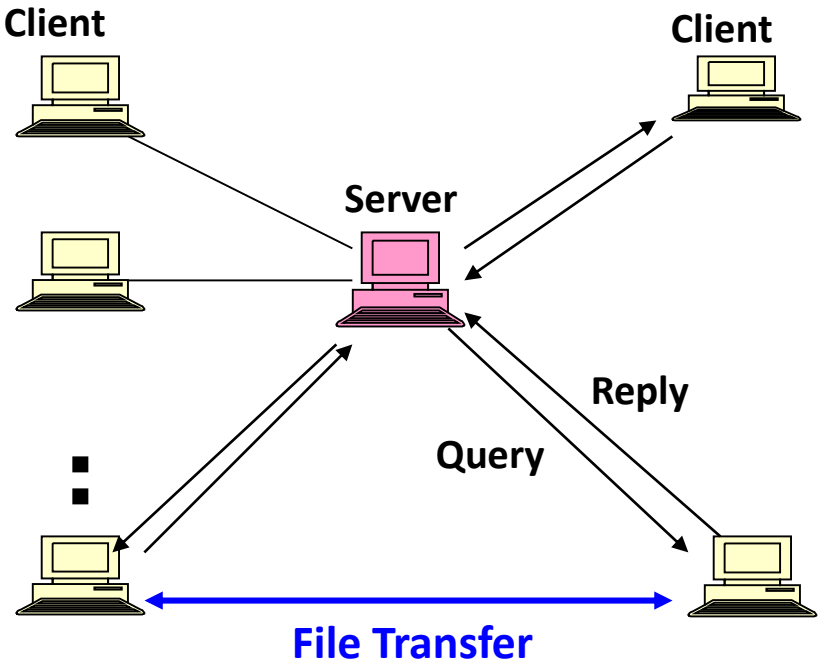
- P2P: concepts & architecture
- Content distribution
- **P2P file sharing**
- Other P2P applications

Napster

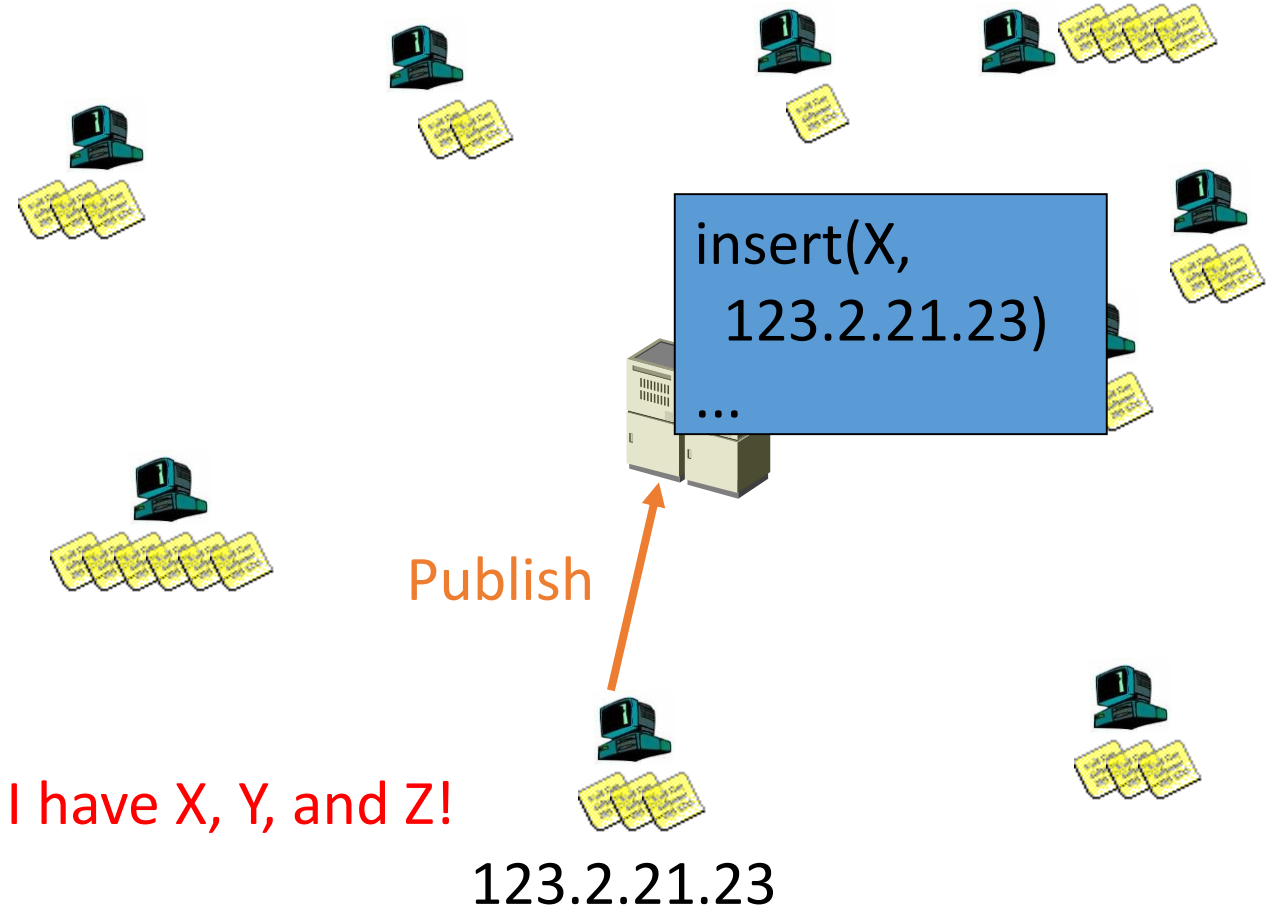
- Centralized lookup: directory service at the Napster server
  1. Connect to Napster server
  2. Upload list of files to server
  3. Give server keywords to search the full list
  4. Select “best” of correct answers.

- Lookup is centralized, but files are copied in P2P manner

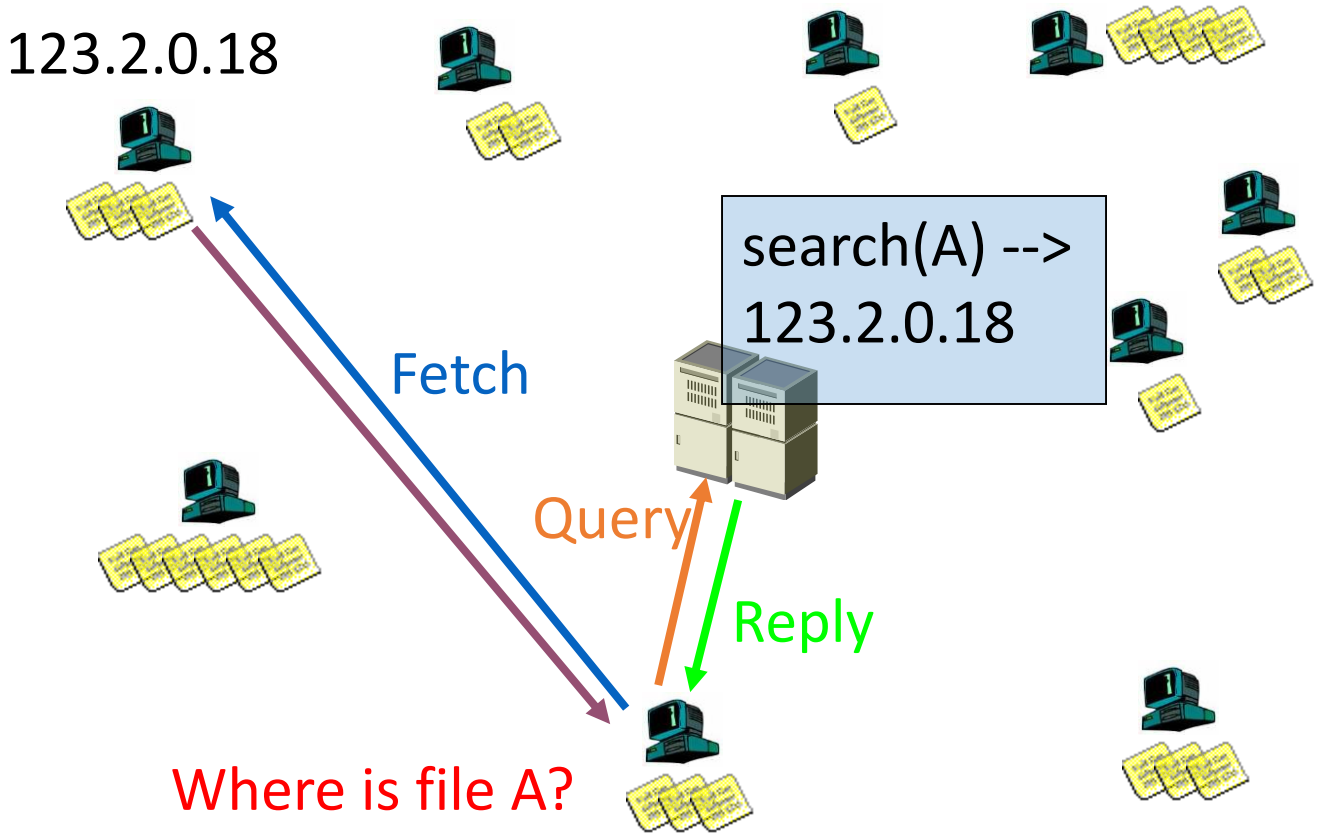
- Napster was shutdown in 2002 due to copyright law issues



Napster: Publish

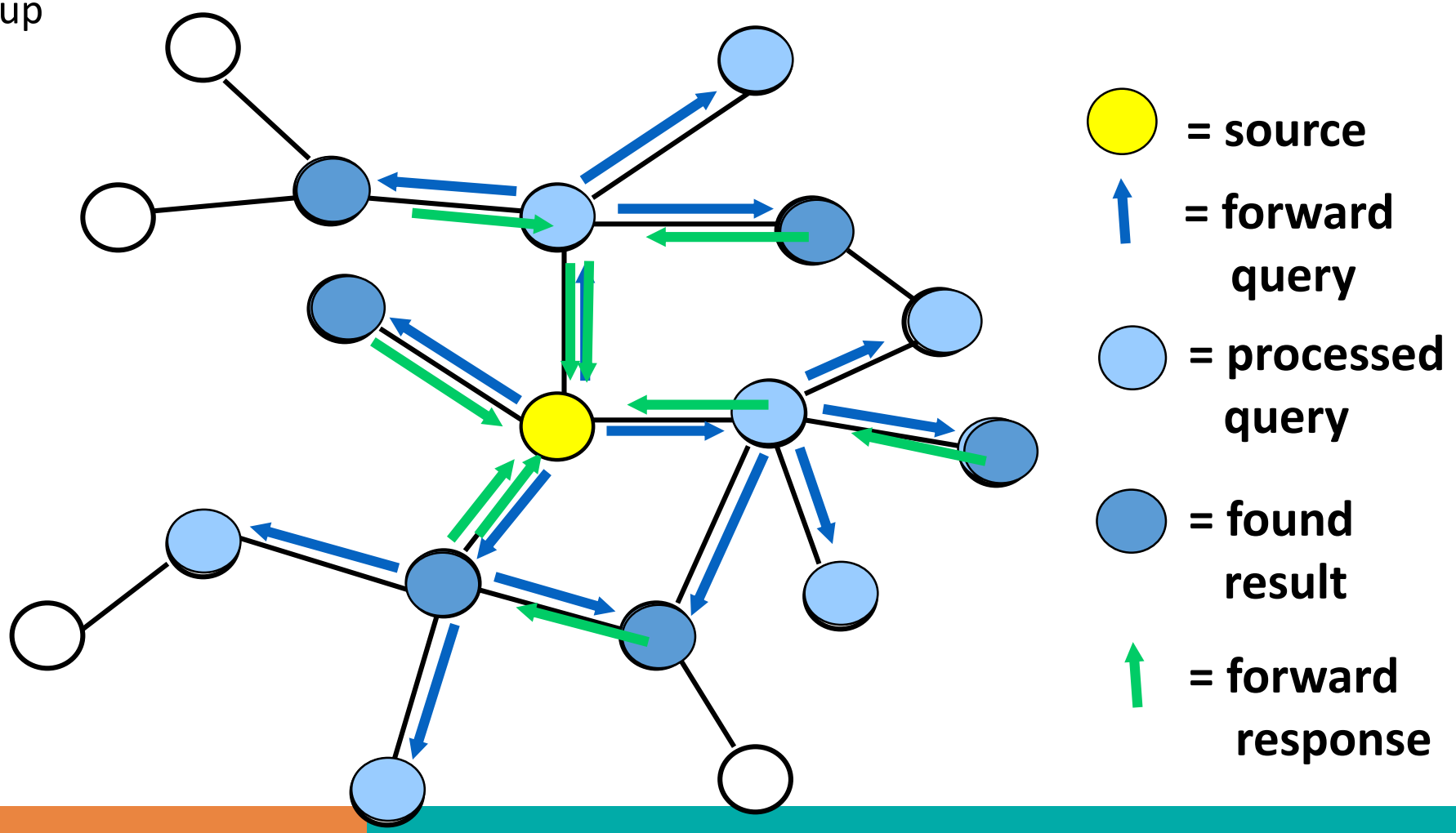


Napster: Search



Gnutella

- The main representative of “unstructured P2P”
  - Flooding based lookup



Gnutella vs. Napster

- **Decentralized**
  - No single point of failure
  - Not as susceptible to denial of service
  - Cannot ensure correct results

- **Flooding queries**
  - Search is now distributed but still not scalable

Gnutella: Query Flooding

南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

- A node/peer connects to a set of Gnutella neighbors
- Forward queries to neighbors
- Client which has the Information responds.
- Flood network with TTL for termination

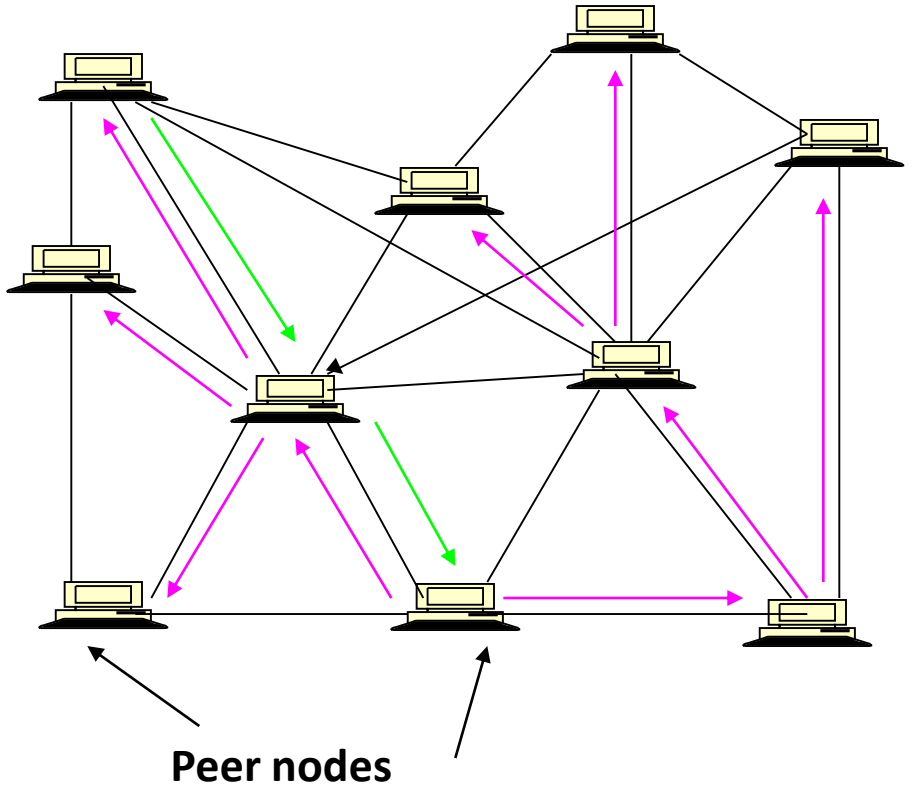
+ **Results are complete**

– **Bandwidth wastage**

Gnutella: Random Walk

南方科技大学 SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

- Improved over query flooding



• Same overlay structure to Gnutella

• Forward the query to random subset of it neighbors

+ Reduced bandwidth requirements

– Incomplete results

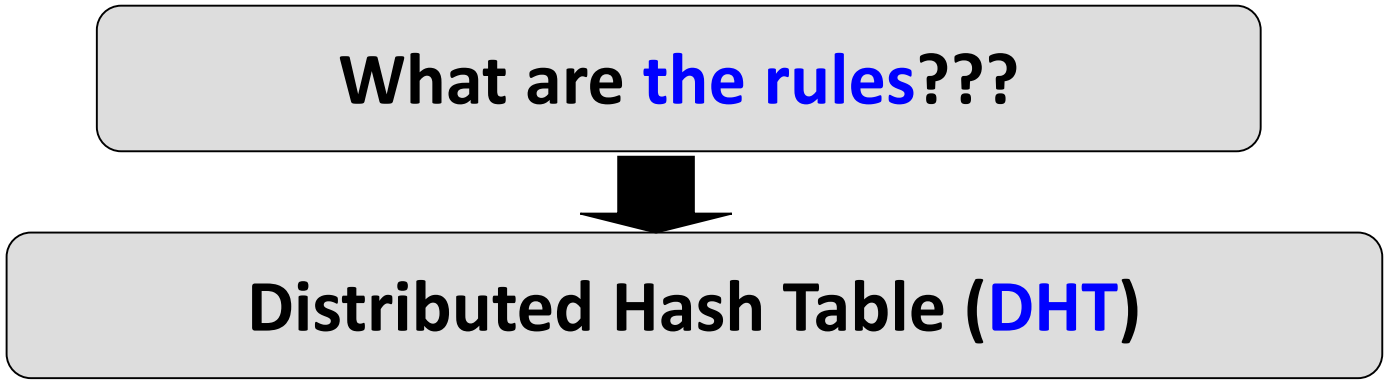
– High latency

Unstructured vs Structured

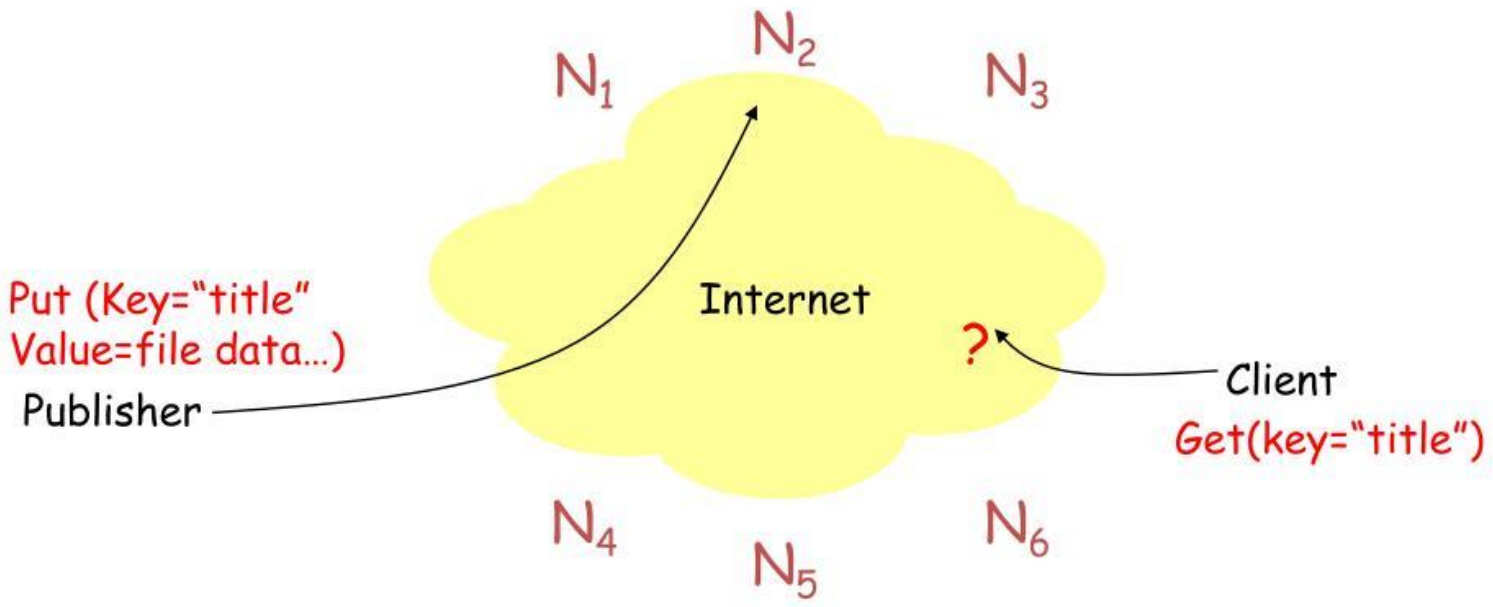
重要

- **Unstructured P2P networks** allow resources to be placed at any node. The network topology is arbitrary, and the growth is spontaneous

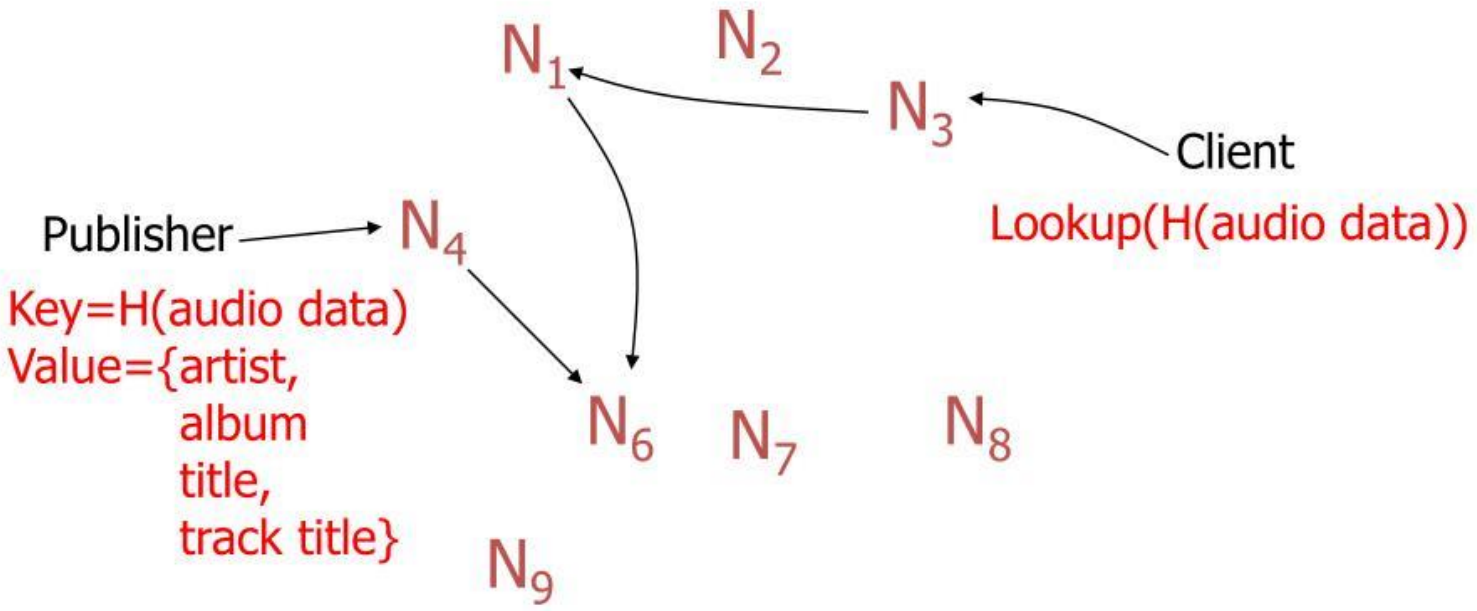
- **Structured P2P networks** simplify resource location and load balancing by defining a topology and defining rules for resource placement
  - Guarantee efficient search for rare objects



The Lookup Problem



DHTs: Main Idea



DHT Overview

南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

- **Abstraction**: a distributed "hash-table" (DHT) data structure supporting two operations
  - put(id, item);
  - item = get(id);

- **Implementation**: nodes in a system form a distributed data structure
  - Can be Ring, Tree, Hypercube, Skip List, Butterfly Network, ...

- **Challenges**
  - Efficient: find items quickly over a large number of nodes
  - Dynamic: nodes join and leave the P2P network frequently (high churn), low relocation of items is needed

DHT Operations

- **Join**: on startup, a node contacts a “bootstrap” node to integrate itself into the network, and it will get a node id
- **Publish**: route publication for file id toward a close node id along the data structure
- **Search**: route a query for file id toward a close node id. Data structure guarantees that query will meet the publication
- **Fetch**: two options:
  - Publication contains actual file => fetch from were query stops
  - Publication contains IP address of the data owner, and get file X from this IP

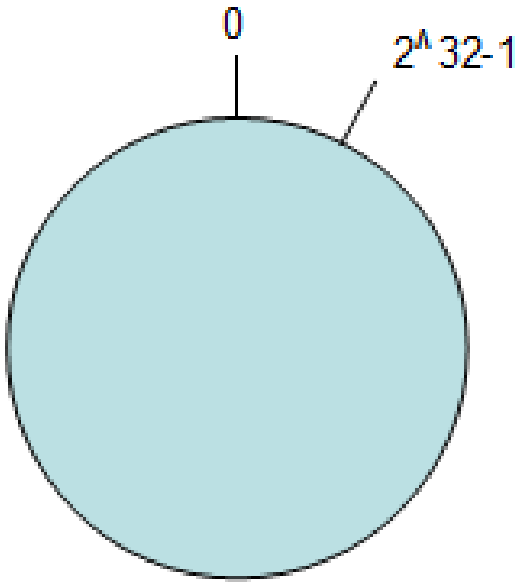
From Hash Tables to Distributed Hash Tables

- Data partitioning problem
  - Given file X, choose one of k nodes to store
- Modulo hashing: place X on node i = (X mod k)
  - Data may not be uniformly distributed
- Basic hashing: i = hash(X) mod k
  - What happens if a node fails or joins (k => k$\pm$1)?
    All mapping keys become obsolete!!

Consistent Hashing

南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

- Hash space in circle for both objects (files) and nodes

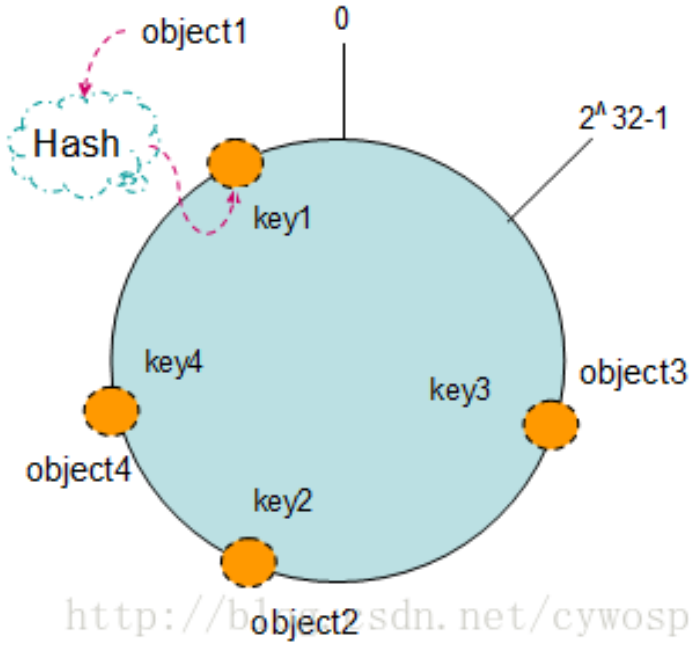


Consistent Hashing

南方科技大学 SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

- Mapping files/objects on to the circle using their hash keys

  - Hash(object1) = key1
  - Hash(object2) = key2
  - Hash(object3) = key3
  - Hash(object4) = key4

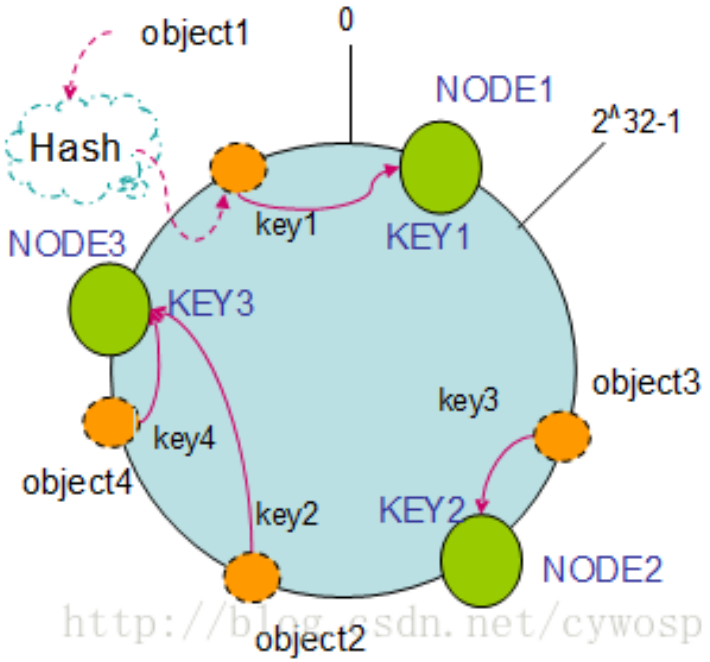


Consistent Hashing

南方科技大学 SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

- Mapping nodes on to the circle using their hash keys

  - Hash(NODE1) = KEY1
  - Hash(NODE2) = KEY2
  - Hash(NODE3) = KEY3



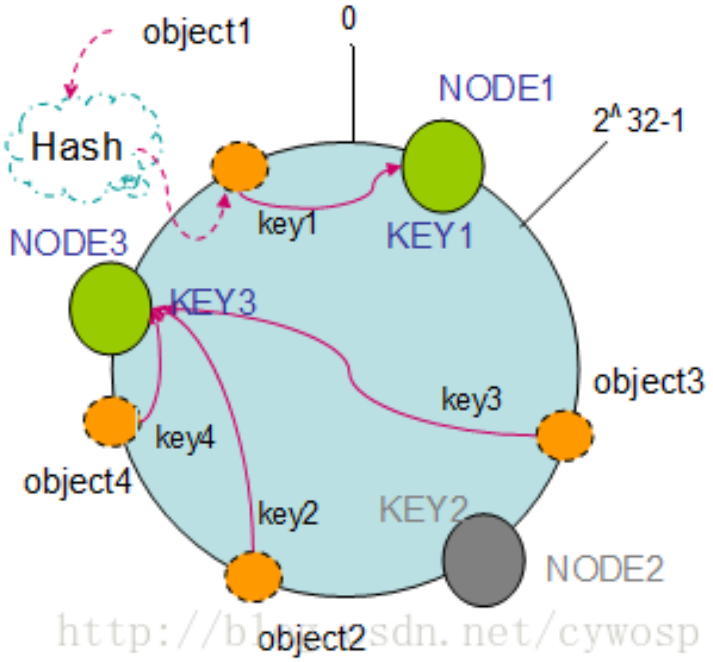
Objects are stored to nodes nearest in clock-wise on the circle

Consistent Hashing

南方科技大学 SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

- Node removal and object remapping

  - Hash(NODE1) = KEY1
  - Hash(NODE2) = KEY2
  - Hash(NODE3) = KEY3

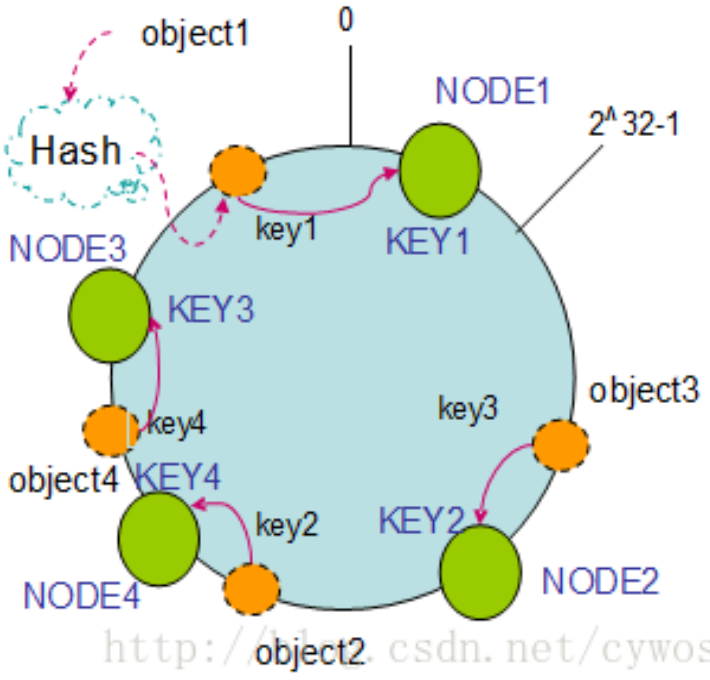


Consistent Hashing

南方科技大学 SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

- Node addition and object remapping

  - Hash(NODE1) = KEY1
  - Hash(NODE2) = KEY2
  - Hash(NODE3) = KEY3



DHT Example: Chord

南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

- Consistent hashing based on an ordered ring overlay
- Both keys and nodes are hashed to 160 bit IDs (SHA-1)
- Then keys are assigned to nodes using consistent hashing (Successor in ID space)

Chord : Hashing Properties

南方科技大学 SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

- **Consistent hashing**
  - **Randomized**: all nodes receive roughly equal share of load
  - **Local**: adding or removing a node involves an O(1/N) fraction of the keys getting new locations

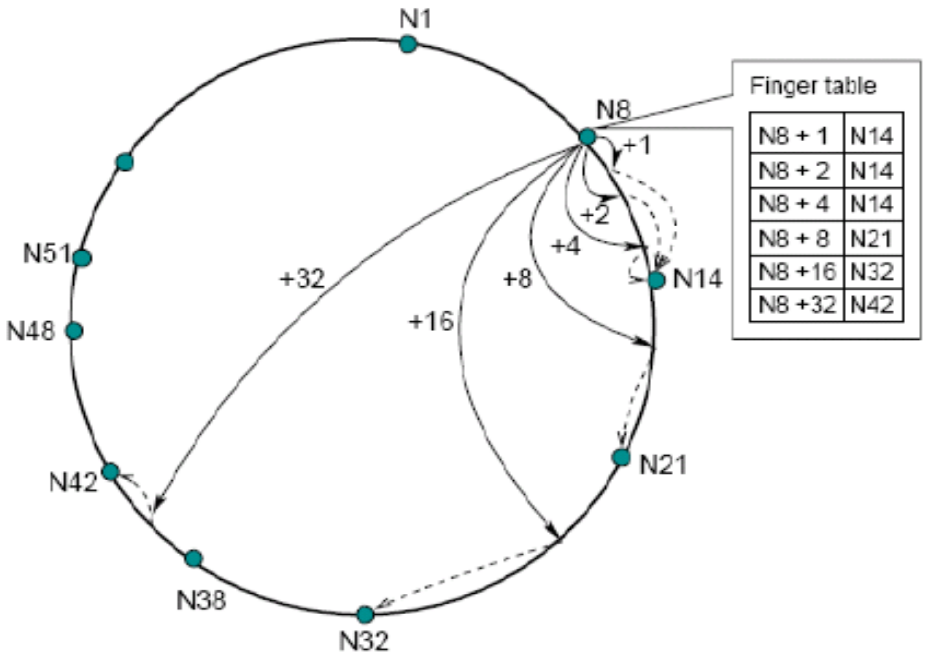
- **Actual lookup**
  - Chord needs to know only **O(log N)** nodes in addition to successor and predecessor to achieve **O(log N)** message complexity for lookup

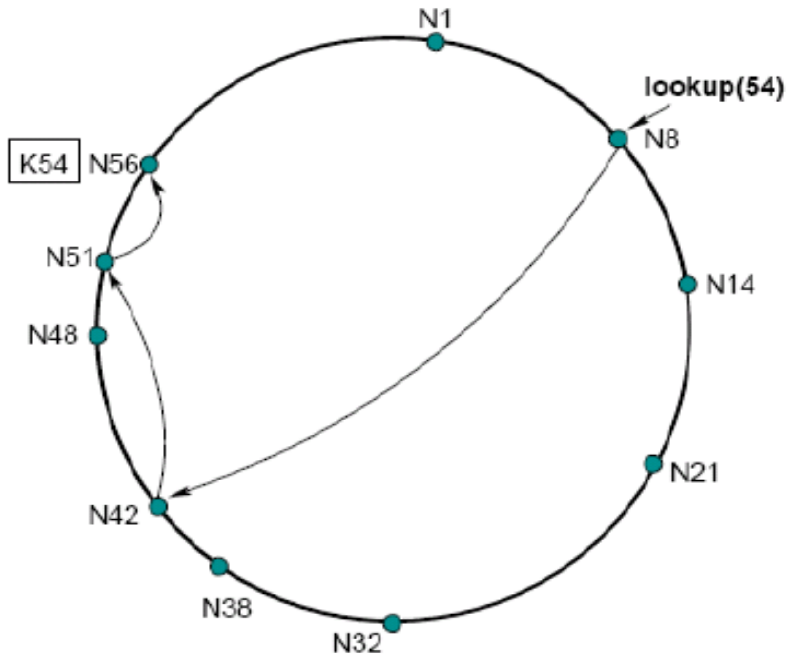
Chord : Primitive Lookup

南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

- Lookup query is forwarded to successor (one way)
- Forward the query around the circle
- In the worst case, O(N) forwarding is required (In two ways, O(N/2))

Chord : Scalable Lookup





i_{th} entry of a finger table points the successor of the key (nodeID + 2^i)

A finger table has O(log N) entries and the scalable lookup is bounded to O(log N)

Chord : Node Join

南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

- A new node has to
  - Fill its own successor, predecessor and fingers
  - Notify other nodes for which it can be a successor, predecessor of finger
- Simpler way : find its successor, then stabilize
  - Immediately join the ring (lookup works), then modify the structure

Chord : Stabilization

南方科技大学 SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

- If the ring is correct, then routing is correct, fingers are needed for the speed only

- **Stabilization**
  - Each node periodically runs the stabilization routine
  - Each node refreshes all fingers by periodically calling find_successor($n+2^i-1$) for a random i
  - Periodic cost is O(logN) per node due to finger refresh

Chord : Failure Handling

南方科技大学 SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

- Failed nodes are handled by
  - Replication: instead of one successor, we keep **r successors**
    - More robust to node failure (we can find our new successor if the old one failed)
  - Alternate paths while routing
    - If a finger does not respond, take the previous finger, or the replicas, if close enough

- At the DHT level, we can replicate keys on the r successor nodes
  - The stored data becomes equally more robust

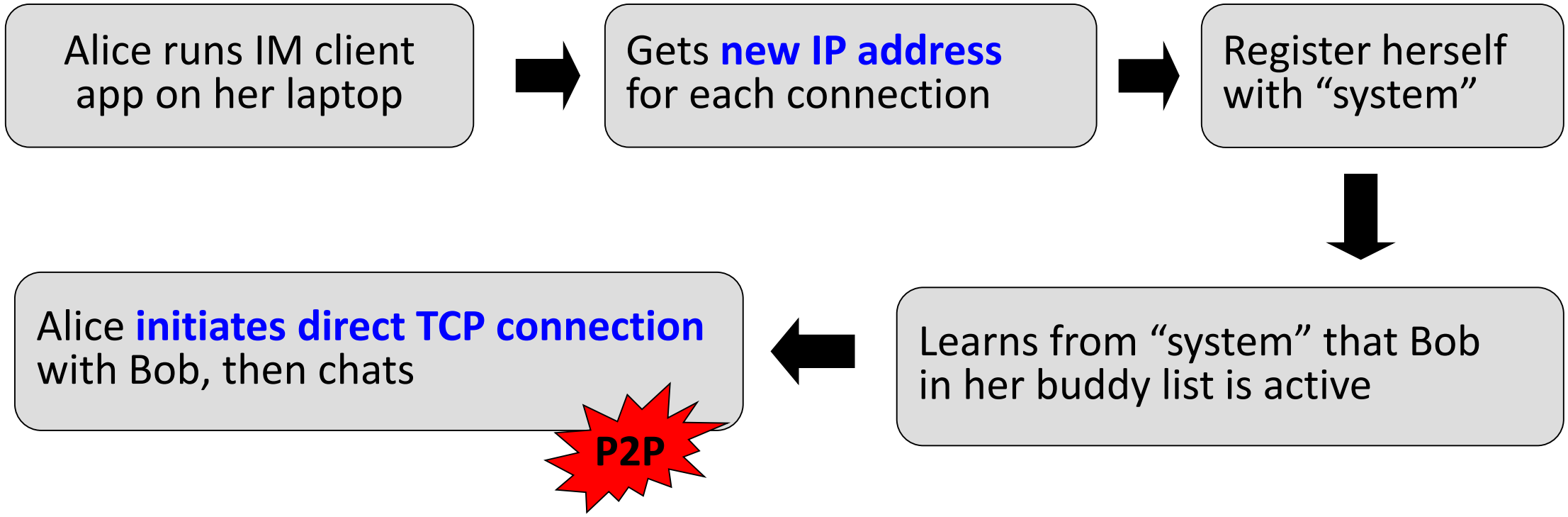
Outline

- P2P: concepts & architecture
- Content distribution
- P2P file sharing
- **Other P2P applications**

P2P Communication

南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

- Instant Messaging
- Skype is a VoIP P2P system

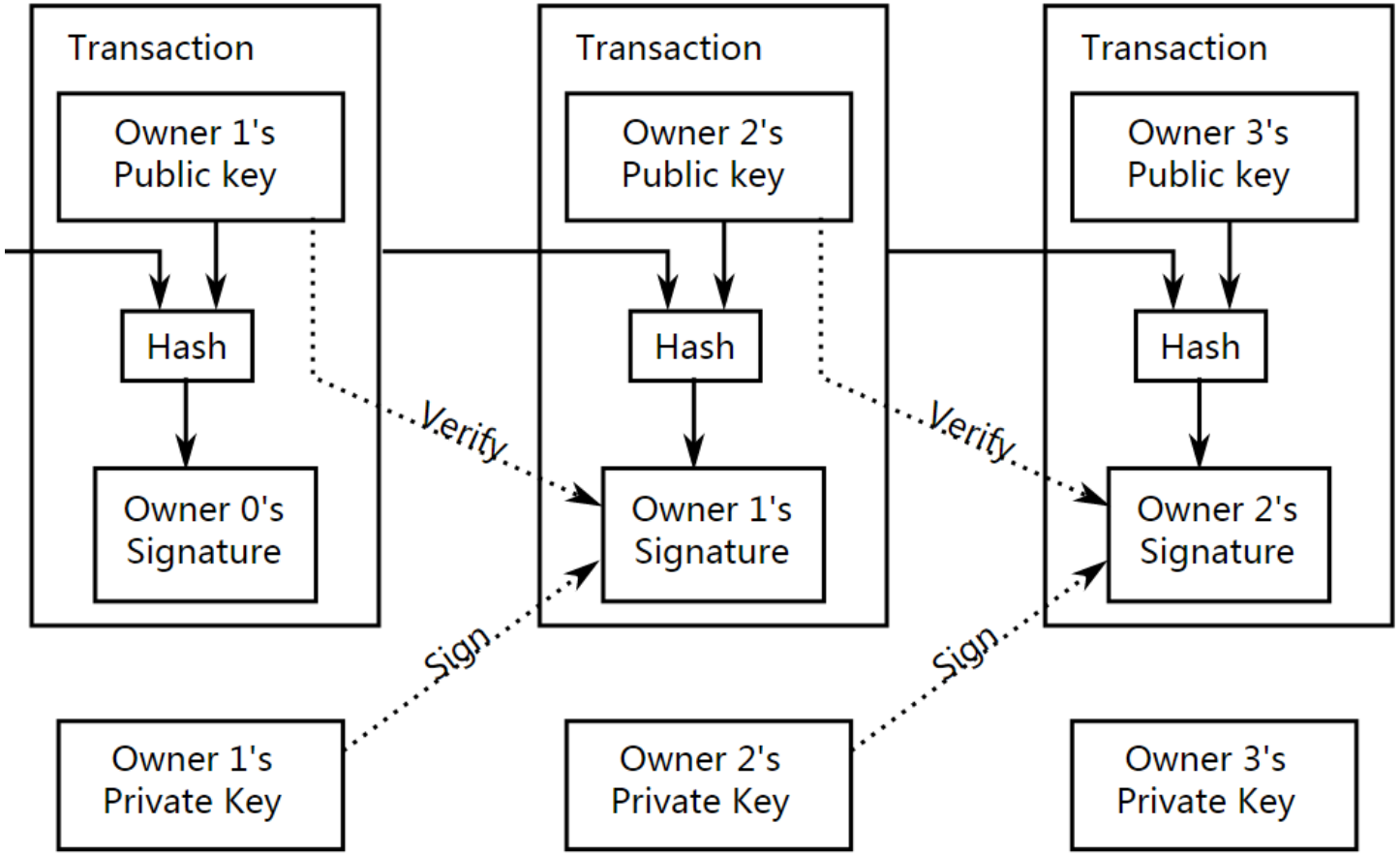


Bitcoin Network

南方科技大学 SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

- A peer-to-peer payment network operating on a cryptographic protocol
- Users send and receive bitcoins by broadcasting digitally signed messages to the network using bitcoin cryptocurrency wallet software
- Transactions are recorded into a distributed, replicated public database known as the blockchain, with consensus achieved by a proof-of-work (PoW) system called mining
- The network requires minimal structure to share transactions. An ad hoc decentralized network of volunteers is sufficient
- Nodes can leave and rejoin the network at will. Upon reconnection, a node downloads and verifies new blocks from other nodes to complete its local copy of the blockchain

A Diagram of Blockchain Transfer

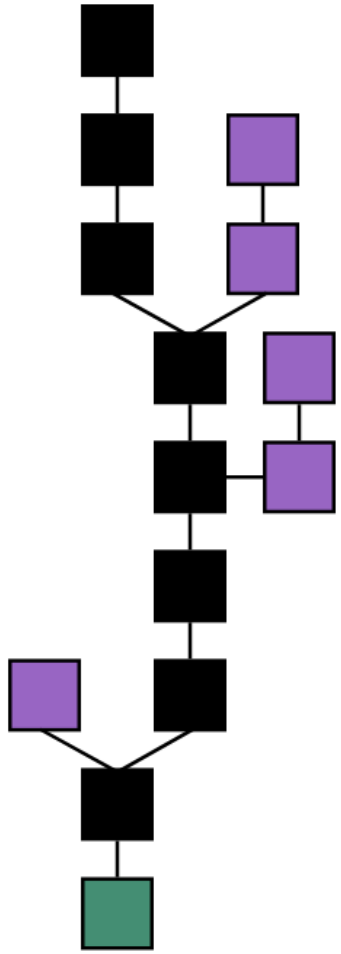


Blockchain based Transaction

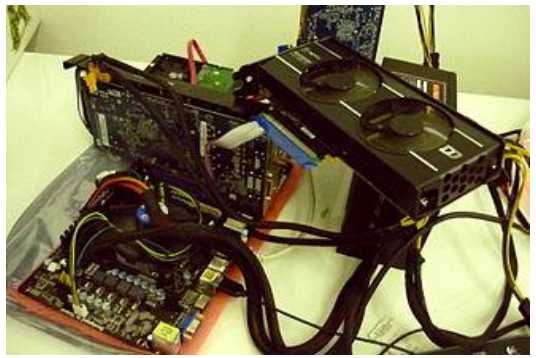
南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

- The best chain consists of the longest series of transaction records from the genesis block to the current block or record

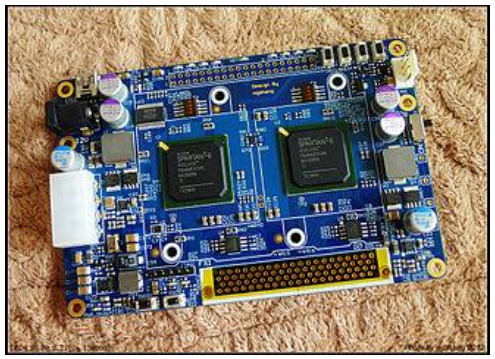
- Orphaned records exist outside of the best chain



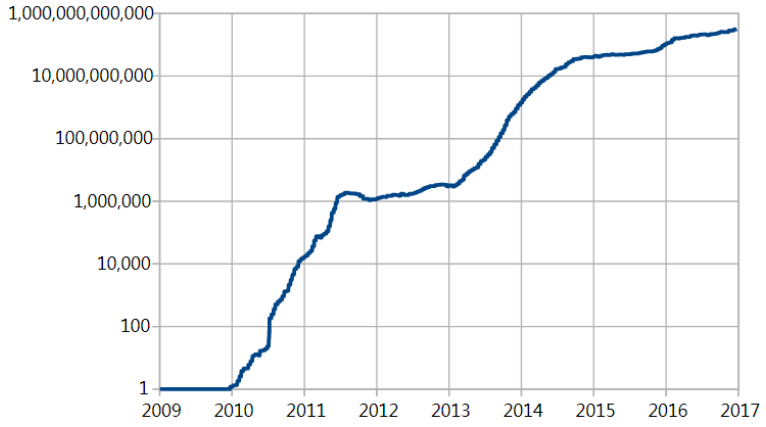
Bitcoin Mining and Transaction Trends



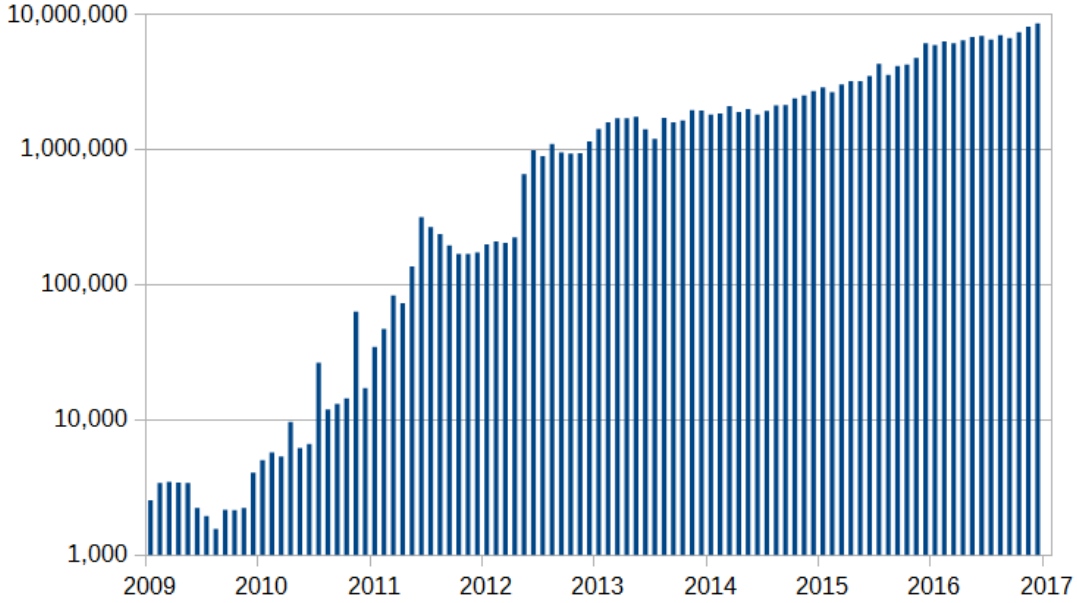
GPU-based Mining



FPGA-based mining



Mining difficulty



Bitcoin Transactions