ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

Information-Centric Networks

Section # 5.2: Content Distribution

Instructor: George Xylomenos

Department: Informatics

European Union
European Social Fund

OPERATIONAL PROGRAMME
EDUCATION AND LIFELONG LEARNING
investing in knowledge society
MINISTRY OF EDUCATION, LIFELONG LEARNING AND RELIGIOUS AFFAIRS
MANAGING AUTHORITY

NSRF 2007-2013
programme for development
EUROPEAN SOCIAL FUND

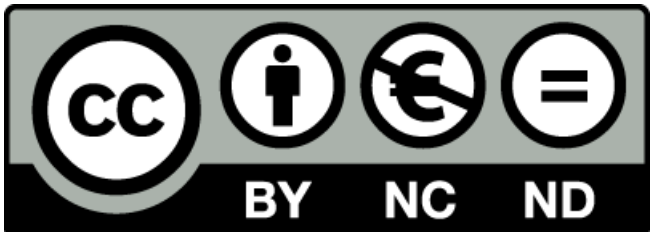
Co- financed by Greece and the European Union

Funding

- These educational materials have been developed as part of the instructors educational tasks.
- The **“Athens University of Economics and Business Open Courses”** project only funded the reformatting of these educational materials.
- The project is being implemented as part of the Operational Program “Instruction and Lifelong Learning” and is co-financed by the European Union (European Social Fund) and national funds.

Licencing

- These educational materials are subject to a Creative Commons License.



Week 5 / Paper 2

- A survey of peer-to-peer content distribution technologies
  - Stephanos Androutsellis-Theotokis, Diomidis Spinellis
  - ACM Computing Surveys, Volume 36, Issue 4, 2004
- Main point
  - Content distribution is one of many uses of P2P
  - Design decisions influence nonfunctional characteristics
  - A taxonomy of P2P systems for content distribution is given

What is P2P?

- Gnutella? Kazaa? Or Napster?
  - From purely distributed to based on centralized servers
- Some emphasize the use of resources at the edge
  - But Napster also does that, although it relies on servers
- Two defining characteristics
  - Sharing of resources by direct exchange
    - Centralized servers only for specific tasks (e.g. bootstrap)
    - Nodes need to participate in maintenance tasks
  - Instability and variable connectivity is the norm
    - Fault-tolerant and self-organizing capacity
- Definition of P2P based on above characteristics
  - Covers Gnutella and Kazaa but not Napster
    - However Napster is also mentioned as many see it as P2P

Classification of P2P architectures

- Communication and collaboration
  - Chat and instant messaging
- Distributed computation
  - Use of CPU cycles from the edge
- Internet service support
  - Multicast, indirection, security
- Database systems
- Content distribution
  - Focus of this study
  - Support data exchange in various ways
    - Direct file exchange between nodes (Gnutella)
    - Shared storage media (Oceanstore)

Content distribution with P2P

- P2P applications
  - Content distribution systems based on P2P technology
  - P2P file exchange systems
    - Facilities to search and transfer files between peers
    - Generally best-effort
  - P2P content publishing and storage
    - Create a distributed storage medium
    - Generally controlled access to the data
- P2P infrastructures
  - Not working applications but substrates for them
  - Routing and location
  - Anonymity
  - Reputation management

P2P object location and routing

- P2P nodes form an overlay network
  - Connections between nodes are paths on a physical network
    - Typically the physical (underlay) network is IP with UDP or TCP
- Overlay network centralization types
  - Purely decentralized: all nodes are the same (servents)
  - Partially centralized: some supernodes are more important
  - Hybrid decentralized: central servers facilitate the network
- Network structure
  - Unstructured: content placement unrelated to topology
    - Content location is tricky (flooding or random walks)
  - Structured: pointers to content are algorithmically placed
    - Only works well with exact-match queries
  - Loosely structured: routing hints to simplify content location

Unstructured architectures

- Hybrid decentralized (e.g. Napster, Publius)
  - Central directory server
    - Holds client info and file table
  - Clients report their files to server
    - The server updates its tables
  - Clients send queries to server
    - The server returns query results to clients
  - Clients exchange files directly
  - Inherently non-scalable, easy to take down
    - It is sufficient to take down the centralized components

Unstructured architectures

- Purely decentralized (e.g. Gnutella)
  - All nodes are equal (SERVer ENTIties, servents)
  - Servents exchange 4 messages
    - Ping: probe to a neighbor
    - Pong: response to ping, contains number and size of files
    - Query: search request (string) to neighbor
    - Query Hit: response to query with matching file pointers
  - First locate some hosts via public server, then send pings
  - Then start sending queries
    - Pings and queries are flooded up to a number of hops
    - Duplicate detection mechanisms also limit flooding
    - Responses are returned via the reverse path
  - Download files directly from the servents having them

Unstructured architectures

- Partially centralized (e.g. Kazaa)
  - Supernodes are dynamically assigned to assist others
    - Supernodes are more powerful than other nodes (CPU, network)
    - Index files shared by peers and proxy queries on their behalf
    - All queries initially sent to supernodes
  - Supernodes make queries much faster
  - But they can be replaced if they fail
- Improving unstructured architectures
  - The biggest problem is scalability
  - Replace flooding by random walks
  - Proactively replicate data
  - Maintain local indices of data

Structured architectures

- Loosely structured systems (e.g. Freenet)
  - Estimate which node is responsible for content
  - Each node makes a local decision to forward queries
  - In Freenet file names are hashed to produce keys
  - There are 4 types of messages
    - Data insert: includes data and key
    - Data request: includes key and TTL
    - Data reply: includes actual data
    - Data failed: cannot find data
  - On insert, each node estimates where similar keys are stored
    - New files are placed close to files with similar keys
    - Each node maintains a table with routing hints
  - On query, local estimates towards possible targets are used
    - Each successful response leaves behind a chain of pointers

Structured architectures

- Chord: uses a ring of nodes and keys
  - Nodes are placed on a ring
  - Each node knows its successor
  - Keys are placed on the same ring
  - Each key is stored at its successor
  - Insertions/deletions of nodes only have local effect
    - Some keys need to move to the next/previous node
  - A finger table is used to speed up lookups
    - Pointers to nodes at exponential distances
    - Allows quick traversal of the ring

Structured architectures

- Tapestry (and Pastry, which is very similar)
  - Nodes and keys are treated as n digit numbers
  - Each node has a routing table with n rows
  - In each row there are pointers to nodes with matching prefixes
    - At row k nodes match the first k digits of the present node
  - Routing proceeds by matching one digit at a time
  - There is some leeway in choosing routing entries
    - Any node with a matching prefix will do
    - Normally choose the closest one
  - Extensions allow replication of data

Structured architectures

- CAN (Content Addressable Network)
  - Multi-dimensional key space
  - Keys and nodes are mapped to points in the space
  - Nodes split the space between them based on proximity
  - Keys are stored at the node within which their point resides
    - Insertions/deletions of nodes require splitting/merging areas
  - Routing proceeds between neighbor nodes
- Improving structured architectures
  - The biggest problem is inexact matches
  - Various proposals for building indexes on top
    - Insert pointers to files with the right keywords as names
  - Maintenance is also quite heavy
    - At least compared to unstructured architectures

Caching, replication and migration

- Replication improves availability and performance
- Passive replication
  - Occurs as nodes exchange content
- Cache-based replication
  - Cache responses passing through the node
- Active replication
  - Push data proactively
- Introspective replica management
  - Observe traffic to decide on replication
- Dynamic replica management
  - If queries are slow, place additional replicas

Security

- Secure storage of data
- Self-certification
  - File names contain a hash of the content
- Information dispersal
  - Encode the data as m blocks and spread them in the network
  - Require any n<m blocks to retrieve the data
  - Can be extended to secretly share keys
- Anonymous cryptographic relays
  - Data is sent to forwarders with send it to storers
  - The publisher destroys the data and publishes the storers

Security

- Access control, authentication and identity management
  - P2P systems are decentralized
  - The same entity can appear with many names
    - This can subvert content replication, routing, etc.
    - Reputation management also cannot work
  - The use of multiple fake identities is called a Sybil attack
  - Some systems use access control mechanisms
    - Only users with specific keys can access content
  - Other systems use challenges to prevent Sybil attacks
    - Need to spend some CPU cycles to be accepted
    - Small numbers of nodes cannot mount large attacks

Anonymity and deniability

- Anonymity
  - May refer to publisher, storer, content or query
  - Dissassociation of content source and requestor
    - Prevent linking files with origin or destination
  - Anonymous connection layers
    - Onion routing and mix routing obscure the routing process
  - Censorship resistant lookup
- Deniability
  - Deniability of stored content
    - Normally based on encryption or steganography
  - Deniability of content in transit
    - Via anonymous connection layers

Incentives and accountability

- Trust-based versus trade-based incentive mechanisms
  - Prevent free-rider behavior in decentralized systems
- Reputation mechanisms
  - Need to propagate local reputation estimates globally
  - Quite tricky in a decentralized environment
  - Many simple mechanisms are easily subverted
- Micropayment mechanisms
  - Implement a micro currency scheme
- Resource trading schemes
  - Nodes issue receipts for services offered
  - Receipts indicate the quality of an entity

Content management capabilities

- Content deletion and update
  - Some systems only support immutable files
  - Others search for and remove old copies of files
- Content expiration
  - Easy way to get rid of stale data
- Content versioning
  - Used in Oceanstore for archival storage
- Directory structures
  - Mnemosyne provides UNIX like directories
- Content searching (see structured vs. unstructured)
- Storage and bandwidth management

ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

End of Section # 5.2

Course: Information-Centric Networks, **Section # 5.2: Content Distribution**

Instructor: George Xylomenos, **Department:** Informatics

European Union
European Social Fund
OPERATIONAL PROGRAMME EDUCATION AND LIFELONG LEARNING
investing in knowledge society
MINISTRY OF EDUCATION, LIFELONG LEARNING AND RELIGIOUS AFFAIRS
MANAGING AUTHORITY
NSRF 2007-2013
programme for development
EUROPEAN SOCIAL FUND
Co- financed by Greece and the European Union