Distributed “Peer-to-peer” Systems

Jasleen Kaur

Fall 2014

Object Discovery Protocols

- How to design a distributed system that helps members share objects (by exchanging copies of objects between peers)?
  - Objects could be files, music, video, …
    - e.g., distributed file systems, usenet, naming service, etc.
  - Object could be available at one, many, or none of the peers

- How to quickly find (and retrieve) the object?
  - Data is important (location isn’t)
    - Query refers to data (and not location)
    - Data placement unrelated to overlay topology
Structured vs. Unstructured

Unstructured Systems

- Trivial overlay construction and maintenance
- Unreliable and random search
- e.g., gnutella

Structured Systems

- Conform to a particular graph structure
- Complex construction and maintenance
- Allow reliable and efficient object location
- e.g., chord, pastry

In the Beginning .... Napster

Issues?
Gnutella – Discovery & Join Flooding

Self-organizing Overlay Network

Gnutella – Search Flooding

HTTP
Gnutella – Rules for Flooding

- Hop-count in messages limits horizon for forwarding
  - May cause search to fail even if target exists
- Incoming Ping or Query messages are forwarded on all connections except the one that sent it
- Incoming Ping or Query messages that are duplicates should be discarded
- Response messages should be forwarded only on the connection sending the original Ping or Query
- Response messages should be discarded if the node has not received the matching Ping or Query

Alternatives to flooding?

Can Gnutella Be Made Scalable?

Random Walk vs Flooding

Issues:
- Termination (TTL vs explicit)
- State maintained
- Number of walkers

Reduces number of messages by two orders of magnitude!
Can Gnutella Be Made Scalable?

Degree of Replication

- Replication should be proportional to square-root of query rate
- Distributed implementation:
  -- # of replicas created should be proportional to length of search

Topography Adaptation

- High degree nodes observe high query loads
- Random walks that seek out high-degree nodes lead to superior scaling to large systems
- Bias overlay organization so:
  -- High capacity nodes have high out-degree
  -- Low capacity nodes are "near" high capacity ones
  -- High capacity nodes are likely to answer query
KaZaA (Fasttrack)

Client → Login Server
Login → Register
SuperNode address → SuperNode
Become SuperNode → SuperNode

KaZaA

Client → SuperNode
HTTP request/response → SuperNode
IP address, port → SuperNode
Who has .....?
I have .....?

Client → SuperNode
IP address, port → SuperNode
Who has .....?
I have .....?