Comparison of BitTorrent with Traditional Content Distribution Networks (CDNs)

ENSC 835 – High Performance Networks

Catherine Chan

Sean Puttergill

Roadmap

- Introduction & Motivation
- Overview of Related Work
- Technical Details
- Implementation & Simulation in ns-2
- References

Motivation

- Traditional CDN
  - client-server based
  - fixed infrastructure
  - network of HTTP and/or FTP mirror sites
  - server to download from selected by locality
  - all upload cost placed on server
  - does not scale
  - not free!

Introduction

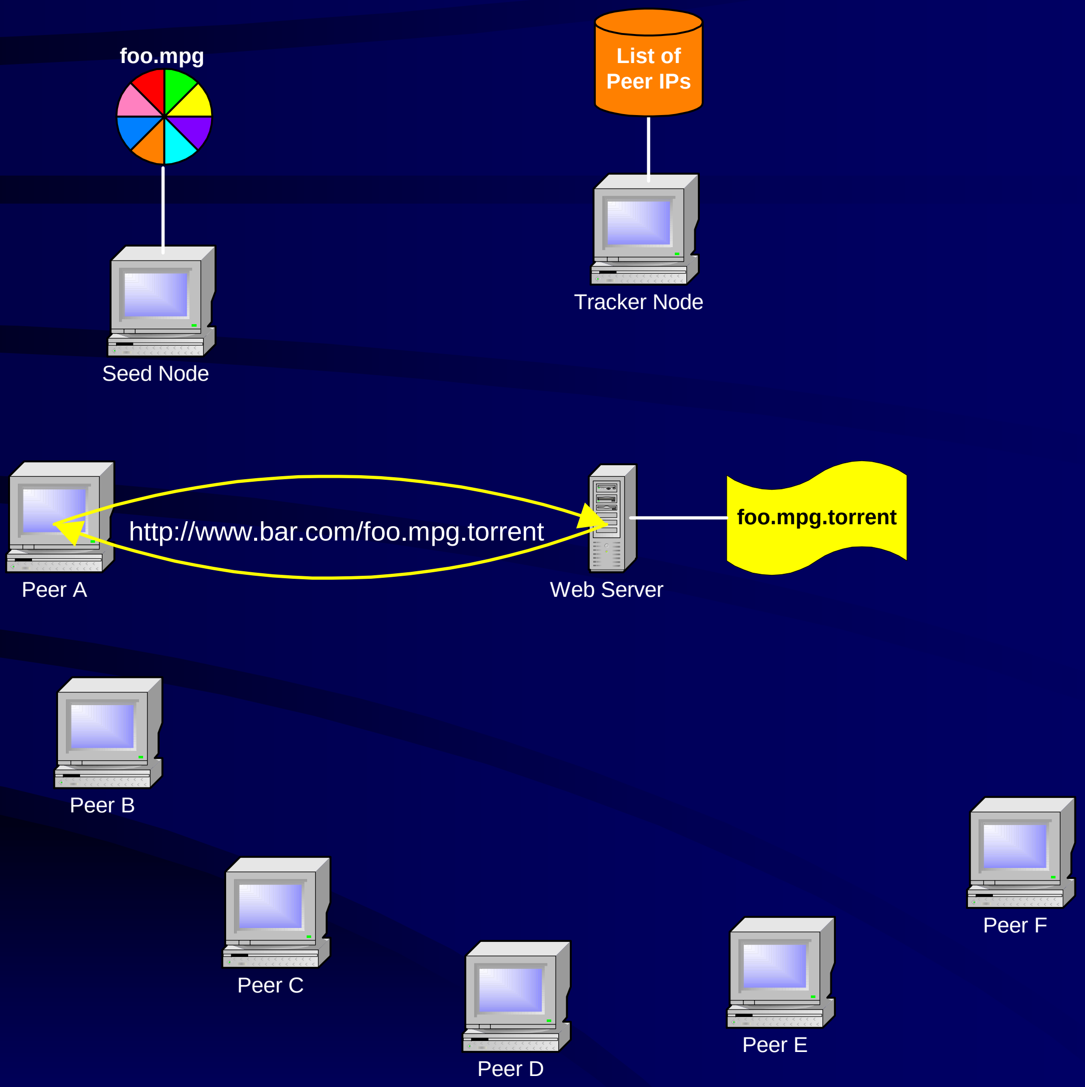
- BitTorrent
  - peer-to-peer (P2P)
  - ad-hoc
  - sophisticated protocol with numerous optimisations to increase efficiency
  - fairness - downloaders required to upload
  - balance upload & download rate through choking

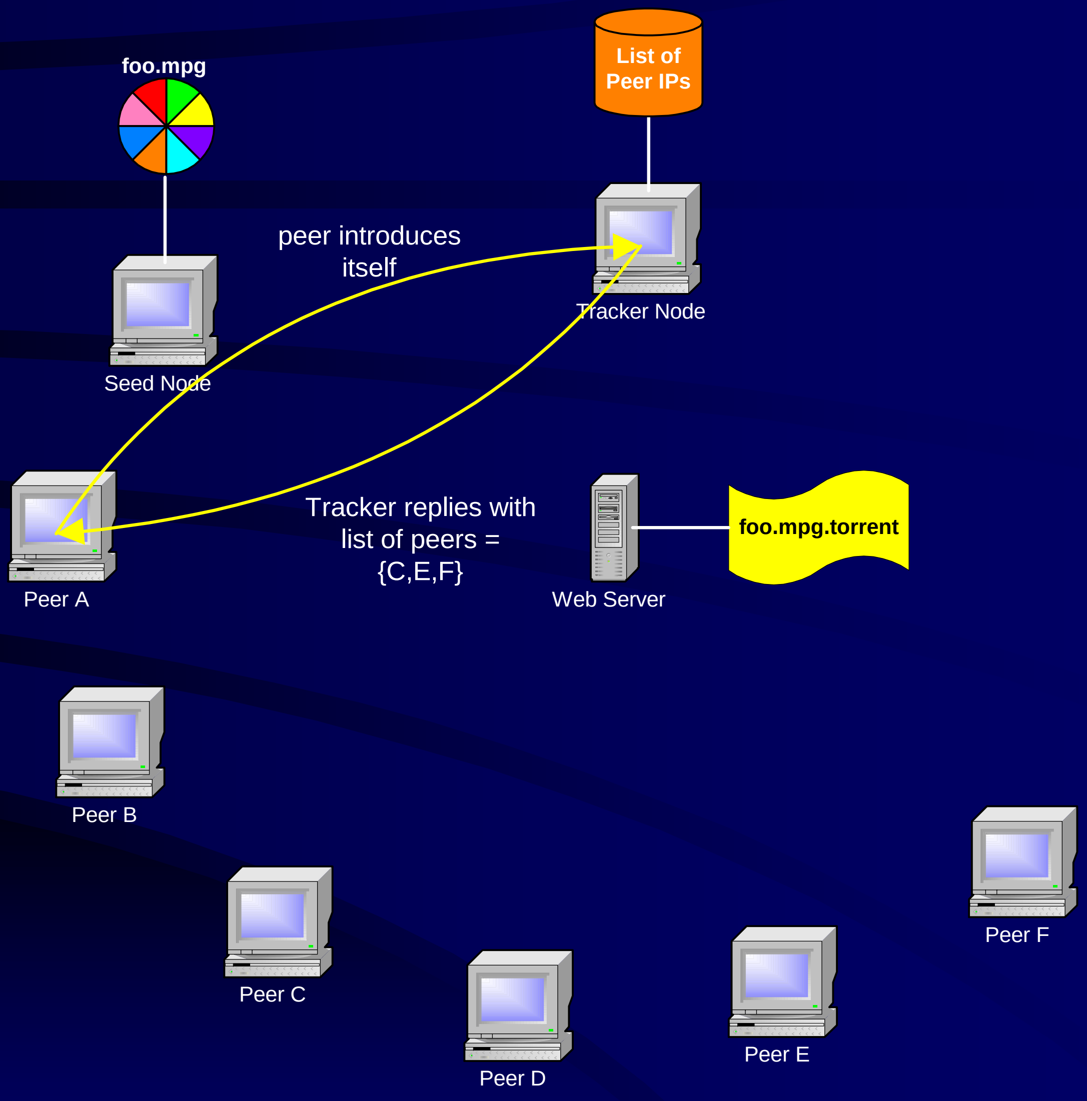
Overview of Related Work

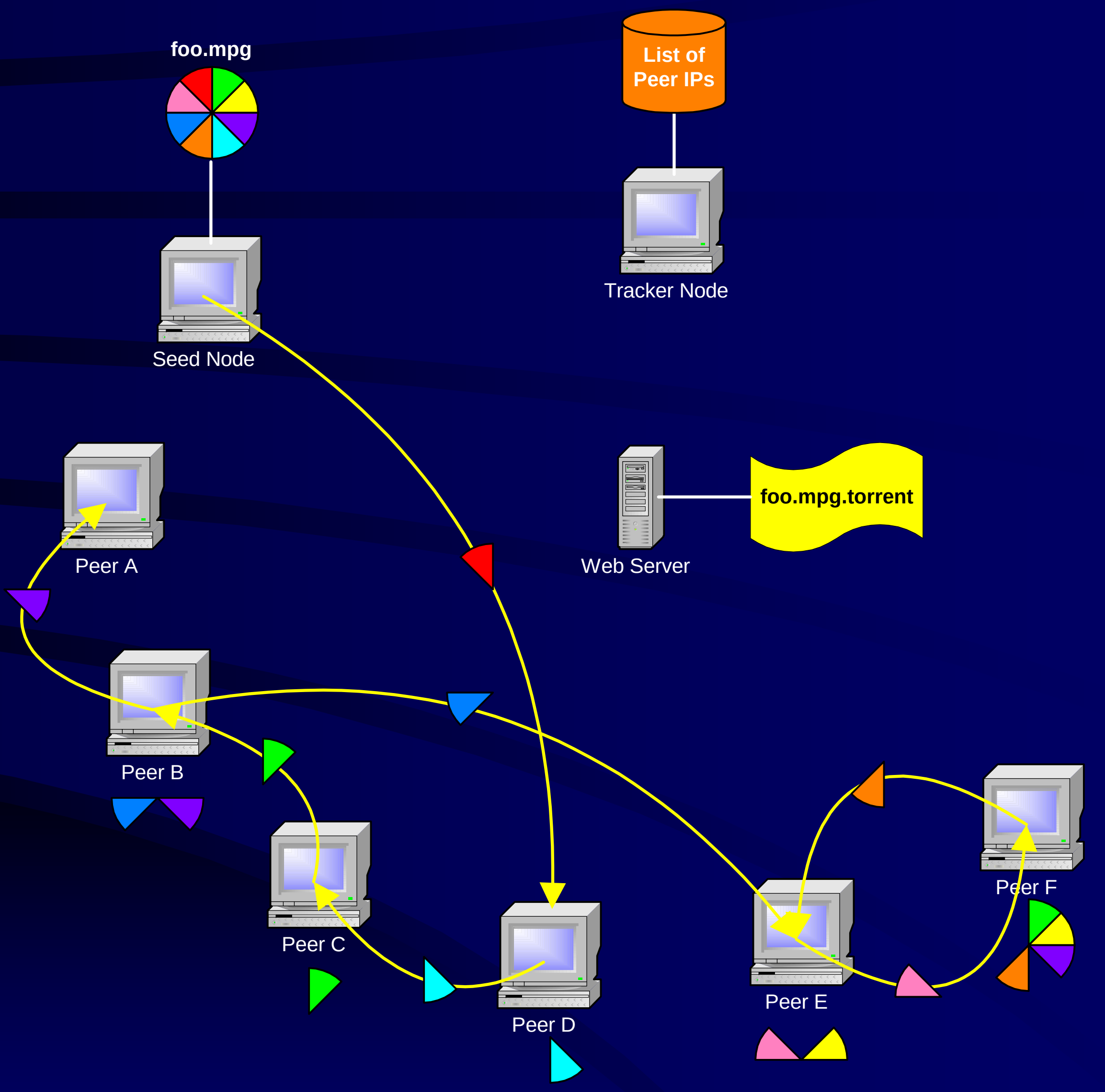
- "Traditional" P2P networks
  - Gnutella, Kazaa/FastTrack, etc.
- Konspire-2b
  - broadcast, random first, copy and forward
- Logistical backbone (L-bone)
  - running on Internet2
  - developed at University of Tennessee
  - file stored in logistical network depots (L-bone)
  - XML encoded metadata file (exNode) maps segments of file to L-bone storage locations

Technical Details

- .torrent metafile
  - contains file info & tracker URL
  - served from regular Web server
- Tracker node
  - maintains list of peers & gathers statistics
  - returns random subset of peers when queried
- Seed node
  - starts off with complete file


foo.mpg
List of Peer IPs
Seed Node
Tracker Node
Peer A
http://www.bar.com/foo.mpg.torrent
Web Server
foo.mpg.torrent
Peer B
Peer C
Peer D
Peer E
Peer F


foo.mpg
List of Peer IPs
peer introduces itself
Tracker Node
Seed Node
Tracker replies with list of peers = {C,E,F}
foo.mpg.torrent
Web Server
Peer A
Peer B
Peer C
Peer D
Peer E
Peer F


foo.mpg
List of Peer IPs
Tracker Node
Seed Node
Peer A
Web Server
foo.mpg.torrent
Peer B
Peer C
Peer D
Peer E
Peer F

Protocol Highlights

- File fragmentation
  - content file split into fixed sized pieces
  - pieces further subdivided into sub-pieces
  - SHA1 hash checks data integrity of pieces
- Pipelining
  - peer keeps multiple sub-piece requests pending
  - offsets overhead of TCP slow-start

Piece Selection

- Peers advertise the set of pieces they have
- How to choose which piece to get next?
- Strict priority
  - always finish a piece before seeking another
  - rapidly obtain complete pieces to share with others
- Rarest First
  - choose rare pieces amongst group of peers
  - helps ensure peer has pieces others want
  - rapidly disseminates pieces from seed node

Piece Selection continued

- Rarest First exception
  - randomly choose first piece
  - get a piece quickly so node has something to share
- Endgame Mode
  - once all remaining sub-pieces are actively being requested then send requests to all peers
  - send cancels as sub-pieces come in
  - prevents a very slow peer with remaining sub-pieces from delaying completion of download

Choking

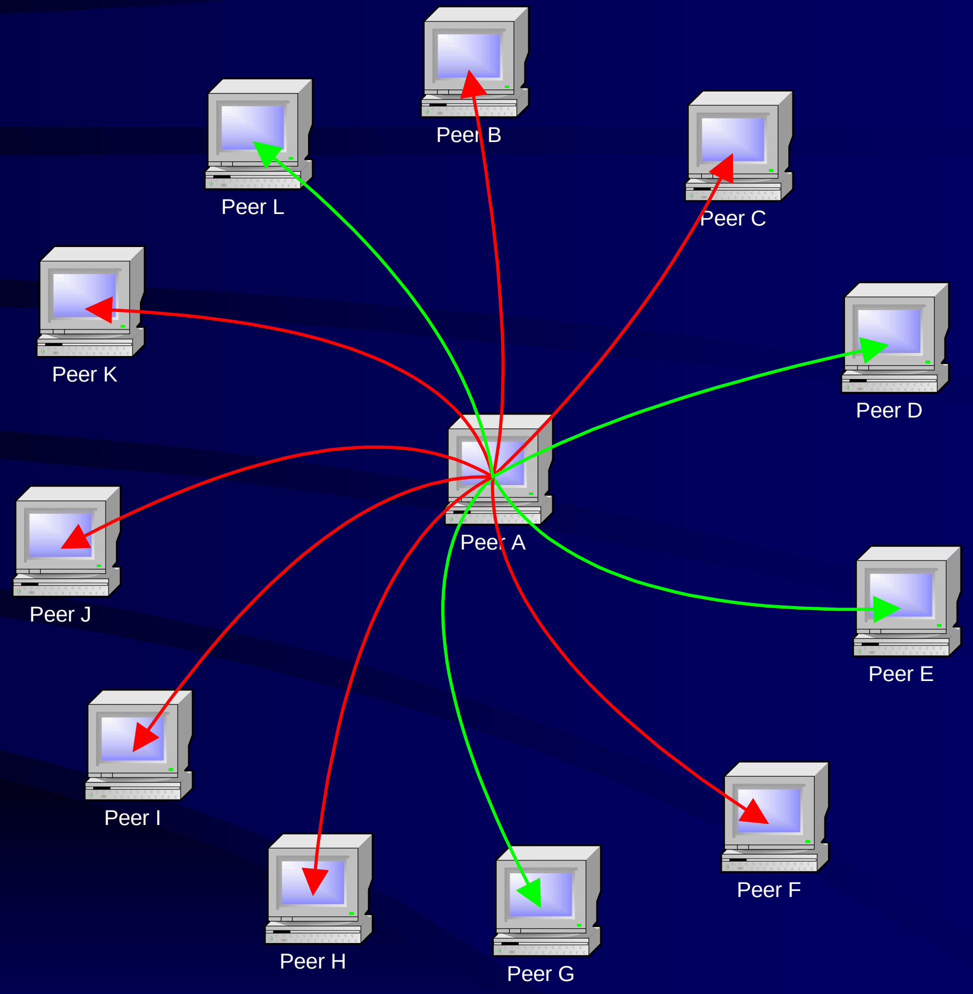
- Enforces balance of upload & download
- Temporary refusal to upload to another peer
- Peers maximise own download rate
- Uses tit-for-tat
  - peers reciprocate uploading to peers who they themselves have successfully downloaded from

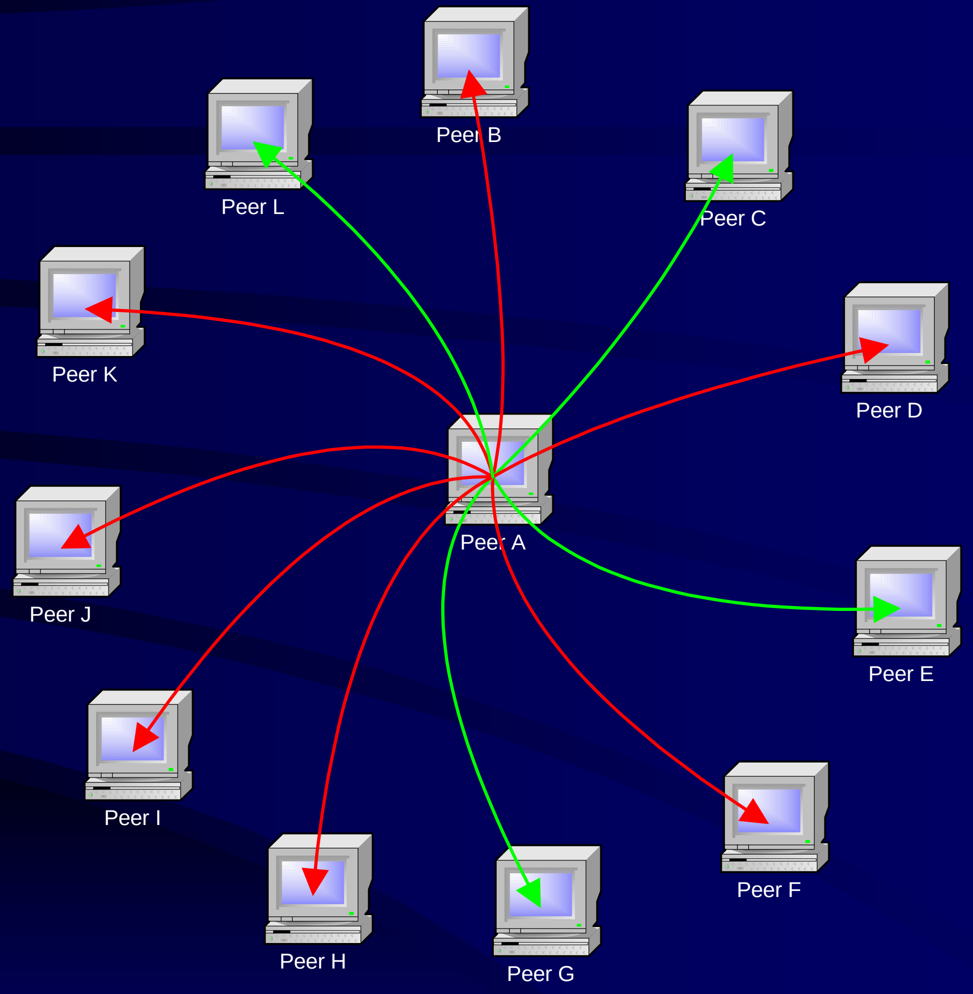
Choking continued

- Peer starts out with all peers choked
- Decides which subset of peers to unchoke
- Choking decisions made every 10s
  - avoids thrashing
- Decisions based on estimated download rate
  - uses 20s rolling average

Optimistic Unchoking

- attempts to find better peers
- periodically replace one unchoked peer with a new peer chosen regardless of download rate history
- optimistic unchoke round robins amongst choked peers


Peer A
Peer B
Peer C
Peer D
Peer E
Peer F
Peer G
Peer H
Peer I
Peer J
Peer K
Peer L


Peer B
Peer L
Peer C
Peer K
Peer D
Peer A
Peer J
Peer E
Peer I
Peer F
Peer H
Peer G

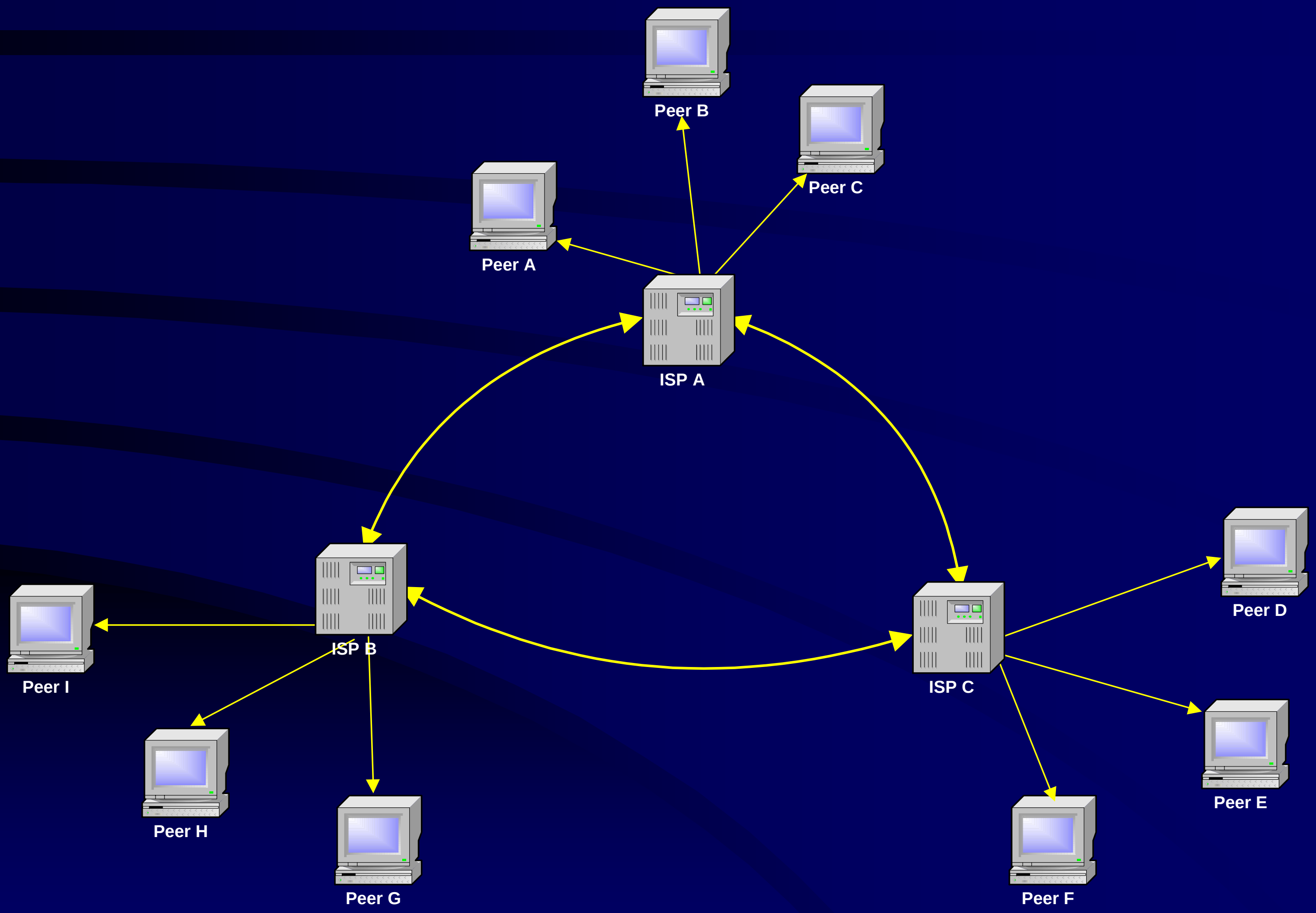
Anti-snubbing

- Occasionally peer will be choked by all peers it was previously downloading from
- If after 1 minute no new pieces obtained then assume snubbed by peer
- When snubbed, stop uploading to peer
- Instead do an additional optimistic unchoke
- Results in faster restoration of download rate

Implementation & Simulation

- Simulate using ns-2 (v2.26)
- Model a traditional CDN
- Model a BitTorrent network
  - implement piece selection and choking
  - abstract other protocol details
- Simulate downloading of single fixed size file
- Metrics
  - bandwidth efficiency
  - elapsed time until all peers complete download

Network Topology



Traditional CDN

- Place CDN server in each AS
- Assume file already replicated on all CDN servers
- Peers download file from closest server
- Download via HTTP

BitTorrent

- Add Seed Node & Tracker Node to network
- Peers require multiple TCP connections so follow GnutApp example
- Peers remain connected for a random amount of time after their download completes

References

1. Bram Cohen, "BitTorrent: Protocol Specification", November 10, 2003. http://www.bitconjurer.org/BitTorrent/protocol.html
2. Jason Rohrer, "Konspire2b: a revolution in mass-scale content distribution", November 10, 2003. http://konspire.sourceforge.net
3. Sir Lancelot, "Bit Torrent Head Quarters", October 23, 2003. http://www.cen.uiuc.edu/~halm/index.html
4. Unknown Author for Webpage, "Wiki theory.org - Bit Torrent FAQ", November 8, 2003. http://wiki.theory.org/index.php/BitTorrentFAQ
5. Sameer G, "A Brand New File Sharing App", November 10, 2003. http://www.rediff.com/netguide/2003/jun/19konspire.htm
6. Guillaume Pierre, "Globule: an Open-Source Content Distribution Network", November 6, 2003. http://www.globule.org
7. A Bassi, M Beck, T Moore, J S Plank, Logistical Computing & Internetworking Lab, “The Logistical Backbone: Scalable Infrastructure for Global Data Grids”, December 2002 http://loci.cs.utk.edu/modules.php?name=Content&pa=showpage&pid=2_4