Comparing Hybrid Peer-to-peer Systems

Beverly Yang
& Hector Garcia-Molina

Presented by L.G. Alex Sung
2nd March 2005 for CS856
Goal of this study

- Maximize UsersPerServer
  - Is it a reasonable performance metric?
  - To be discuss later…
Contributions

• Presents several architectures for hybrid systems
• Presents and evaluates a probabilistic model for queries
• Compares architectures quantitatively, based on their models and data from the music-sharing domain
• Compares strategies in non-music-sharing domains
Batch vs Increment

- **Batch**
  - **Whole** library info add/remove from server
  - High login/logoff network cost
  - Low CPU cost when returning query results

- **Increment**
  - **Incremental update** to library info on server
  - Low login/logoff network cost
  - High CPU cost on filtering active results
Architectures

• Chained
  – Local server attempts to satisfy the query first
  – Forward query to remote servers until certain number of results are found
• Unchained
  – Satisfy query by the local server only
• Full replication
  – Global info stored on each server
• Hash
  – Metadata words hashed to different servers
Max Users vs Query/logon

- Query is more expensive than logon operations
Hardware requirements

- Incremental strategies require more memory and CPU cycles
Overall results

![Graph showing the maximum users supported per server versus the number of servers. The graph includes different line styles and markers for various scenarios like batch CHN, incr. CHN, batch FR, incr. FR, batch HASH, and incr. HASH. Each line represents a different scenario, and the graph shows how the number of users decreases as the number of servers increases.]
Results

• Chained
  – best architecture for the music domain

• Full replication
  – might be good with cheap memory and stable network connections

• Incremental logins
  – best when there is negative correlation between $f$ (query selection power) and $g$ (query popularity)
  – performs best in short, bandwidth-limited sessions
The world today

• The peer abilities enhanced..
  – 56Kbps dial-up ➔ broadband
  – From Mp3s ➔ Mpg movies ➔ DVDs

• But peer selfishness haven’t changed
  – Free-riders still common!

• Bittorrent
  – The 2\textsuperscript{nd} generation P2P applications
Bittorrent highlights

1. Tracker file (.torrent)

2. Random Peer IPs

3. Exchange

Web server

Tracker server

Active peers
Start downloading…

- Get 40 random peers from tracker
- Wait for some of the peers to send you the first piece of data
  - optimistic unchoking
- Pieces passing on to later generations

Tit-for-Tat exchange

• Upload to top 4 peers with max d/l rate
• Optimistic “unchoking”
  – upload some pieces to random peers
  – See what’s the d/l rate from them
• Reset the top 4 peers
• Optimum Strategy:
  – Offer \textbf{max} upload rate
Research problems

• Anonymous BT
  – eXeem (a commercial product w/ ads)
• Non-random peer group distribution
  – Based on content availability [1]
  – Or downloading/uploading speed
• How to best utilize the slow-uploading-fast-downloading bandwidth?
• Performance modeling[2]
Discussion of the paper

What I like?

• Very first work on performance evaluation of P2P systems.
• Potentially the only server performance work with real data for Hybrid P2P Systems in the music sharing domain
• Challenges today
  – How to get real server data without sharing copyrighted materials?
Discussion (cont’)

• Maximizing UsersPerServer is a reasonable metric
• Recall: login ➔ query ➔ download
  – How about response time?
  – More user ➔ longer response time?
• Non-music sharing domains…
  – Only the correlation of $f$ and $g$ is studied.
  – Not representative: Too many differences
References
