BitTorrent Optimization Techniques
(from various online sources)
Announcement

• No recitation next week!
• Final review session
  – Next Sunday (5/2) 5-7pm, GHC 4215
  – Let us know what you want at http://www.doodle.com/6qvsnubhmam2zkxp
  – More specifics will be announced on the course webpage.
Announcement (2)

• TA Evaluations!
  – Your comments / feedbacks are welcomed
    • Any reasonable criticism
    • Anything you liked or didn’t like
    • Anything you would like to do / see

  – Helps us improve the recitations and our teaching style!
Evaluation links

• 441 A  Kaushik Lakshminarayanan
  http://www.surveymonkey.com/s/3WJKKHTM

• 441 B  Rui Meireles
  http://www.surveymonkey.com/s/3WQSLXV

• 441 C  Daegun Won
  http://www.surveymonkey.com/s/3WSD2VW
Before we start

• Everything we discuss here is about BitTorrent
• Not everything might be useful for the project
  – We’d like to make it your work to figure it out
  – It wouldn’t be hard 😊
Two Important Aspects

• Peer selection
  – How to choose other peers to exchange data with

• Chunk selection
  – How to choose / prioritize chunks to download
Peer selection

• Employs a ‘Tit-for-Tat’ strategy
  1. Unless provoked, the agent will always cooperate
  2. If provoked, the agent will retaliate
  3. The agent is quick to forgive
  4. The agent must have a good chance of competing against the opponent more than once.

• Called choking algorithm
Good choking algorithm

• Caps the number of simultaneous uploads
• Avoids choking/ unchoking too quickly
• Reciprocate to peers who
  – let the peer download
  – try to use unused peers once in a while (get out of local maxima)
More specifically...

- Peer A *choke* Peer B if it decides not to upload to B
  - Choking = temporary refusal to upload

- Each peer (A) *unchokes* at most 4 peers that have chunks that A doesn’t have
  - 3 peers with fastest upload rate (to A)
  - 1 randomly chosen peers
    - Called ‘Optimistic Unchoking’
Optimistic Unchoking

- Finds potentially faster peers
- Allow new peers to receive their first piece
- Helps out ‘snubbed’ users
  - Snubbed users = Choked by all its peers
Chunk Selection Strategies

- Random First Piece
- Rarest First
- Endgame Mode
**Chunk Overlaps**

- **Small overlap**
  - Every pair can exchange something
  - Better utilization of bandwidth

- **Big overlap**
  - Only a few peers are very ‘valuable’
  - Less utilization of bandwidth
What do we want?

• Ultimately, we want to maximize the total transfer rate of all simultaneous transfers

• It would be nice if every pair has something to exchange
  – So that we can utilize most of the possible end-to-end connections
What does it have to do with chunk selection?

• If something is …
  – Too popular
    • So much supply but not many looking for it
  – Too rare
    • So much demand but not many having it

• Then it is much less likely to utilize all end-to-end connections
Maximizing Bandwidth Utilization

- Keep all chunks as evenly popular as possible
  - So that we can maximize the number of simultaneous transfers
Prioritizing algorithm should aim towards uniform distribution!
Chunk Selection Strategies

- Random First Piece
- Rarest First
- Endgame Mode
Random First Piece

• Initially, the peer has nothing
  – Important to have some pieces to reciprocate for the choke algorithm.

• Need something ASAP
  – Randomly chosen chunks are likely to be more replicated
    • Can download them faster
  – Then the peer can upload something

• First four piece, then switch to Rarest First
Rarest First

• Look at all chunks at all peers
• Request the ‘rarest’ piece
  – Owned by fewest peers

• Yes, aim towards UNIFORM DISTRIBUTION!

• What if the original seeder leaves before no one downloads the whole file?
  – Oh no!
Rarest First

• What if the original seeder leaves before no one downloads the whole file?
  – This policy increases the likelihood that everything is still available!
Endgame Mode

• Happens near the end
  – Request the missing blocks to everyone.
  – Cancel pending requests when the chunk is downloaded

• A bit wasteful, but…
  – Speeds up the completion
  – Not too much waste in practice
  – Prevents slow completion due to a single peer with slow transfer rate
Anything else you can do?

There are more things you can improve...
Other things you can do

• You can easily improve your congestion-avoidance algorithm
  – Check out other algorithms such as TCP new-Reno *(highly recommended)*

• The network topology might be not totally random
  – May have a number of clusters and so on…

• And of course, look up for more on the web!
Sources

- [http://www.ict.kth.se/courses/ID2210/lectures/Lecture08-BitTorrent.pdf](http://www.ict.kth.se/courses/ID2210/lectures/Lecture08-BitTorrent.pdf)
  - If the above doesn’t work, try [http://docs.google.com/viewer?a=v&q=cache:tW513GkEkXkJ:www.ict.kth.se/courses/ID2210/lectures/Lecture08-BitTorrent.pdf+bittorrent+lecture+pdf&hl=en&gl=us&pid=bl&srcid=ADGEESgjwWMdTDIEQMQWdRkCax3kOlhy4GKfRk-rlhFlhVfP6x5dCyDsIkSisQDKv6lquNIMyCldED-VlsocV1l9k1AGftgzXlOgYbp7IozD2xTeR3WtLiN7Ta93o67o7Y5TL_RF&sig=AHIEtbSh2BxqAKN8ck4kEj-p6yg-j_DcTA](http://docs.google.com/viewer?a=v&q=cache:tW513GkEkXkJ:www.ict.kth.se/courses/ID2210/lectures/Lecture08-BitTorrent.pdf+bittorrent+lecture+pdf&hl=en&gl=us&pid=bl&srcid=ADGEESgjwWMdTDIEQMQWdRkCax3kOlhy4GKfRk-rlhFlhVfP6x5dCyDsIkSisQDKv6lquNIMyCldED-VlsocV1l9k1AGftgzXlOgYbp7IozD2xTeR3WtLiN7Ta93o67o7Y5TL_RF&sig=AHIEtbSh2BxqAKN8ck4kEj-p6yg-j_DcTA)