P2P file sharing lecture

History

- Every 5-6 years, the Internet gets turned on its head
  - email: first killer app
  - ftp: next one
  - Web and CDNs: next
  - P2P file sharing: previous
  - TV through the Web: current
  - Mobility / geolocal / ???: next?
- P2P – ignited by music download
- Confluence of technical trends made music download possible
  - Effective perceptual compression, hard drive capacity, edge bandwidth
  - 1996 was the tipping point [256 kb/s broadband, .mp3]
- Music industry was too slow to move
  - Fear, uncertainty, technical misunderstanding
  - File-sharing systems filled the gap
- File sharing architecture chosen to solve a number of problems
  - No commercial providers – provider bandwidth scarce
    - Need to leverage peer upload bandwidth
    - Technically a rich problem area
      - Load balancing / routing
      - Replication
      - Search
  - Legal question issues
    - Centralized service is a legal target
    - Distributed harder to shut down
- Led to two separate research questions
  - what is this new media workload, and is the Internet capable of serving it?
  - what else is a peer-to-peer architecture "good for" besides stealing music?

Legalities

- Two types of infringment
  - direct – you copied/sold/performed something
  - indirect or "secondary" infringement -- you sell a product or run a system that encourages direct infringement
- Direct infringement
  - not much to say about this, besides "fair use" exceptions exist
    - fair use – broad guidelines and case-by-case exceptions
    - seems likely that file sharing is not fair use in USA
- Indirect infringement – two main theories of it – contributory and vicarious infringement
- Contributory:

- 1. There was some direct infringement
- 2. Defendant knew of should have known about it
- 3. Defendant materially contributed to direct infringement
- Technical defense against contributory – knowledge of infringing acts, and material contribution to infringing acts
  - P2P: only know in general; courts think this is enough
  - Material contribution: if not running server, is up in air
- A key defense against contributory infringement – betamax
  - Defeat if show product is capable of "substantial" or "commercially significant" non-infringing uses
  - Betamax: time travel as a fair use
- Does P2P fall under betamax defense?
  - Answer: so far, yes, with caveats.

- Vicarious:
  - 1. There was some direct infringement
  - 2. Defendant had right or ability to control infringer [police]
  - 3. Defendant derived direct financial benefit from infringing acts
  - Key example:
    - swap meet operator and bootleg tape booth
  - Does P2P fit criteria for Vicarious?
    - answer: 1 and 3 clearly. 2 is the big issue
    - Interestingly, legal liability comes down to technical question of control
    - different answers for different architectures!!
- Recording and movie industry responses
  - 1. Sue toolsmith [so far, bittorrent has escaped – why? No search!]
  - 2. Sue individuals [is "working"]
  - 3. Get congress to change the law
    - new theories of secondary infringment
      - anti-betamax: massive infringement
      - "could have designed it to police"
      - responsibility to implement effective technical filtering
- Judicial system
  - so far, doing the right thing. quote from 9th circuit
  - recognizes "quicksilver technological environment"
    - courts ill suited to fix flow of internet innovation
    - new technology always disruptive
    - time and market forces often bring new equilibrium
  - "inducement" law
    - cannot encourage
  - soon – filtering requirements
    - technological battlefield once again

Technical considerations

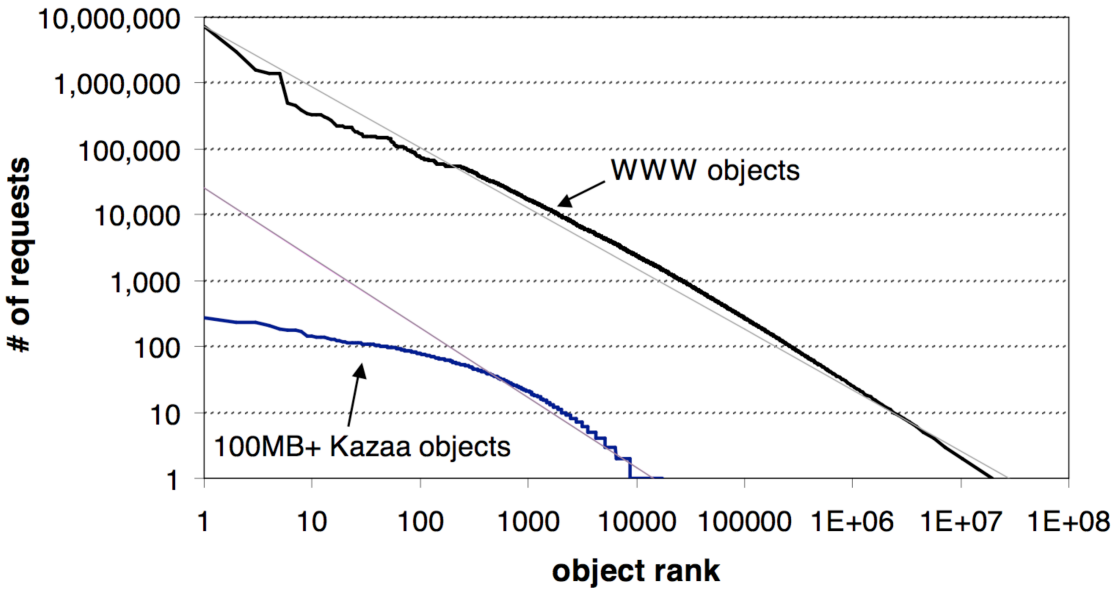
- Three main aspects need to understand
- (1) Characteristics of peers – bandwidth, latency, availability, etc.
    - building blocks
- (2) Workload presented by peers – popularity distribution, file size, etc.
- (3) Architecture of system

Characteristics of peers – MMCN '02 paper

- Substantial heterogeneity in every aspect of peers
    - bandwidth: multi-hump CDF
        - many modems, broadband, and "rich" peers
        - most of hump is in broadband nowadays
        - estimates: several 100 million broadband hosts in Internet
    - latency: wide spectrum from any point in world
        - really three "zones"
            - same coast – O(10ms)
            - opposite coast -- O(100ms)
            - different continent – O(300ms)
        - if traversing an overlay to find content, need to avoid overlay links spanning continents or coasts
    - activity:
        - some "heavy hitter" peers that download a lot
        - some "kernel" peers that upload a lot
        - most peers do neither
            - and most requests are from non-heavy-hitters
            - two real issues here:
                - churn in stable population [try and leave]
                - those that stay decay into steady-state – only fetch the new interesting stuff
    - availability:
        - heavy-tailed availability distribution
        - median time is in short # of hours
        - implications for storage system?
        - is this fundamental?
    - "ethics":
        - freeloading is rampant
            - most people take more than they give if allowed
            - does this matter?
        - lying is common
            - people will misreport bandwidth toward their favor
        - enforcement – how in decentralized system??
            - currency – how to bootstrap, how to make non-forgeable
            - reputation – cheap to falsify, especially with collusion
            - best answer seems to be pairwise fair-exchange

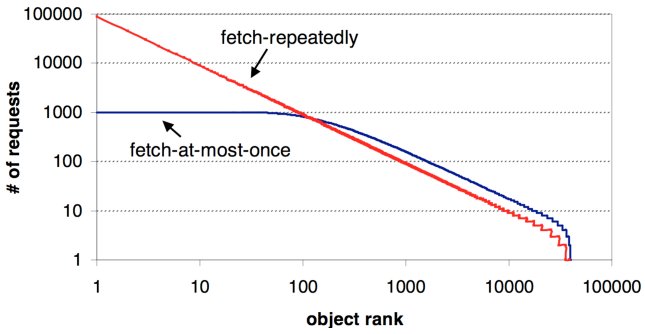
- major conclusions:
  - plenty of resources in peers: bandwidth, storage capacity, CPU
    - but hard to tap
  - even though goal was P2P, ends up looking a lot like client/server
  - "churn" in population leads to lots of issues for durable storage applications
  - heterogeneity in population leads to lots of optimization possibilities for routing applications

Characteristics of workload: SOSP '2003



- popularity: Zipf-like, but with flattened head
  - Zipf: universal property of most human consumption
    - heavy-tails usually a function of reinforcement
    - Popular get half requests, unpopular get other half
      - or something like that – 80/20 rule?
    - head of curve is "easy"
      - widely replicated, easy to find, plenty of local copies, plenty of swarm opportunity
    - tail of curve is "hard"
      - not widely replicated, have to look long and hard to find, will transfer from far away over network, hard to build a swarm
  - Zipf is normally a static notion
    - for P2P, evolution of popularity over time matters just as much
    - typical popularity over time curve
      - height of popularity in youth
      - timescale of days or weeks, depending on format

        - major job of peer-to-peer system – widely disseminate the rising stars.
        - unlike web cache, no strong notion of steady-state warm cache – constant churn of bringing new stars in and kicking old has-beens out
    - Popularity observed is modulated by caching
        - P2P: fetch-at-most-once because objects are expensive and immutable
        - Leads to significantly flattened head
        - reduced opportunity for caching?



- object size:
    - media: 4MB songs, 1GB movies
        - 4MB: about 1 second to download at full bore broadband, ~10 seconds at UL == DL
        - 1GB: about 5 minutes at full bore download, and ~1 hour at UL == DL
        - interesting inflection point: last byte latency < playback time. Allows for streaming media delivery. We're finally there; catalyzed services like Hulu and Netflix VoD.
        - big technical question:
            - should media be stored at home (TiVo + bittorrent)
            - or in the network/ISP (Hulu / VoD)
- degree of replication and "object availability"
    - function of architecture
        - broadly speaking, popular are everywhere, unpopular nowhere
        - Zipf says both parts of distribution matter, though models like Netflix may be changing this
    - high degree of locality because of popular replication
        - in UW trace of kazaa, most bytes (85%) could have been served locally if we had a campus proxy cache

Overall

- pretty different than Web system
    - Zipf vs. Zipf-like
    - Large media objects vs. small text/graphics --- bulk/batch vs. interactive
    - Popularity over time shift

Architecture

Compare on two axis:

1. Search architecture
2. Transport architecture

- Napster
  - centralized directory for search
  - peer-to-peer single-connection download
  - limitations?
    - centralized directory as scaling bottleneck --- nope!
    - single-connection download as transport bottleneck – yup!
    - no "locality-awareness" in selecting peer/servers – backbone pressure
      - no caching architecture – why?
- Gnutella [unstructured]
  - overlay with broadcast for search
    - hugely inefficient and non-scalable
    - popular stuff is findable
    - unpopular is not
    - evolved over time to have metadata caches
  - essentially single-connection download
    - evolved over time to "swarm"
- Fasttrack / Kazaa [semistructured]
  - two-level hierarchy – supernodes and nodes
  - search: search peer neighborhood on supernode first
    - if miss, gnutella amongst supernodes
    - in principle supernode level could be bottleneck, but worked up to millions of nodes
    - find popular fast, unpopular is harder
  - transport: simple swarm
    - MD5 hashes to identify same file at multiple providers
    - user could select "group" of files and download in parallel from each
      - file would be "chunked" into 1-2MB chunks
      - data naturally flows from faster provider
    - no locality awareness
      - backbone pressure
    - no caching
      - organizational pressure
    - no explicit replication
      - unpopular hard to find

Bittorrent: [no structure to finding content -- model is more like multicast]

- No search – you need to name the torrent stream
  - name tracker
- Tracker maintains list of all peers currently in stream
  - each peer can ask for list of random 50
  - contacts up to ~10 in that list and starts asking for blocks
- notion of a "root seed"
  - just a peer that provider puts up that has all the blocks
- "tit-for-tat" protocol
  - peer: only upload to somebody who is uploading to you
    - and from time to time give a freebie
  - purports to solve the freeloading problem
    - ensures download rate is tied to upload rate
    - but degree of freedom in:
      - which peers to "unchoke" and upload to
      - how to split upload rate
      - bittryant: greedy choices lead to better local performance. Not yet clear what global impact is.
- questions:
  - what is the best download performance you should expect from a p2p system?
    - can argue that UL == DL is one natural equilibrium
    - getting more than that means
      - (a) many people are uploading but not download; or
      - (b) any excess you get is stealing from somebody else; or
      - both.
  - does performance get better as more people join the system?
    - no – still get UL == DL, except for a few transients
  - how should root seed be provisioned?
    - some choices:
    - (a) push as much as possible; wasteful
    - (b) constantly push max UL; only helps high bandwidth
      - notion of leverage
    - (c) hope there is enough benevolence out there that just need to maintain at least one copy of each block
      - seems to work in practice – light fuse and run

Long term questions

- can edge bandwidth handle it?
  - currently architected for download, not upload
  - risk of saturation
  - 10-15 year innovation cycle on edge; some countries ahead of us
- if legal issues go away, does P2P still make sense?
  - Web → CDN: pushing content closer to edge
  - not clear it needs to be pushed to other side of edge
  - more of an economic / market force question
- longer term: time-shifting devices changing the nature of media consumption
  - right now is the most traumatic time for content production
  - broadcast: conflates efficient delivery with scheduled consumption
  - Internet: can it have efficient delivery?
    - not clear yet; designed as point-to-point system
    - maybe good [necessary?] for unpopular part of tail
  - In world of unscheduled consumption, what is right architecture?
    - where are bytes stored? [home? everywhere else?]
    - where are they consumed?