Advanced Computer Networks

P2P Swarming

Jianping Pan
Summer 2015

Paper reading summaries

- R1 due this Thursday, June 11
  - reading list is on connex (will be further updated)
  - R2--4 submission schedule to be posted soon
  - 1-page reading summary on assigned papers
  - what's the problem? important then? and now?
  - main ideas? previous work? follow-on work?
  - strengths then? and now? why?
  - weaknesses then? and now? why?
  - how to improve? how to do better or differently?

* reading guideline at: http://www.cs.uvic.ca/~pan/csc466/reading.txt (template available too)

Review: going P2P

- Client-server
  - server is well-known and serves all client requests
  - scalability issue
- Peer-to-peer
  - structured or unstructured
  - every peer is a (potential) server
    - search is a challenge
  - one request is still served by one peer
    - until the peer fails, then try to use another peer

Napster and Gnutella

- Napster
  - Ĥ centralized directory server
    - list uploading and query handling
  - Ĥ peer-to-peer file download
- Gnutella
  - Ĥ fully distributed
    - scoped flooding search
  - Ĥ peer-to-peer file download
- Improving Gnutella
  - Ĥ node hierarchy
  - Ĥ non-flooding search

Q: more design space to explore?

More design choices

- If more than one peer can serve, why do they not serve the same request together?
- Benefit
  - Ĥ more resilient to node dynamic
    - does not rely on any particular peer
  - Ĥ fit better with the asymmetric access link
    - higher download bandwidth than upload
- Overhead
  - Ĥ how to get served from multiple peers
    - work together constructively

The BitTorrent approach

- Chop a file into small, fixed-size pieces
  - e.g., pieces (usually 256 KB each)
  - and then into blocks (usually 16 KB each)
- .torrent
  - meta information about the file
  - out-of-band retrieval
- Tracker
  - return a list of peers may have some pieces
- Seed and leecher/downloader
  - peers have the complete/incomplete file

.torrent

- Tracker URL
- File info
    - name, length
- Piece info
    - length, hash
- Other info
    - date, comment, etc
- Bencoding
    - strings, integers, lists, directories
    - e.g., 4:*spam*, i*3*e, l*4:spam4:eggs*e, d4:spaml*1:a1:b*ee

Tracker protocol

- HTTP GET request
  - info_hash: to identify the file
  - peer_id: of the requesting peer
  - client address and port: to respond to incoming requests
  - bytes uploaded, downloaded, left, etc
  - numwant: the number of peers in the response list
- Tracker response
  - failure reason, if any
  - contact interval
  - peer list and stat (seed and leecher, etc)
- Tracker-less mode (on Kademlia DHT)

Tit-for-tat

- Download while upload: tit-for-tat
  - upload to whom from which download: trading pieces
  - prevent free-riding
    - fairness?
- Choking/unchoking
  - a limited number of uploads
    - default: 4+1
  - evaluate peers based on their recent download speed
    - 20-second average
  - upload to the peers with the fastest download speed
    - adjust every 10 seconds

Optimistic unchoking

- Stuck with poor peers?
- Optimistic unchoking
  - Ĥ upload to other peers as well
    - rotate every 30 seconds
  - Ĥ hope to get better download
  - Ĥ also help bootstrap other peers
- Seed's unchoking
  - Ĥ seed does not download from other peers
  - Ĥ try to equally distribute its upload to leechers
  - Ĥ or upload to the one downloads fastest

Peer wire protocol

- Messages over TCP
  - handshake
  - keep-alive
  - choke/unchoke
  - interested/not-interested
    - a block is downloaded if the client is interested and unchoked
    - a block is uploaded if the peer is interested and unchoked
  - have
    - advertise new pieces
  - request/piece
    - request blocks in a piece

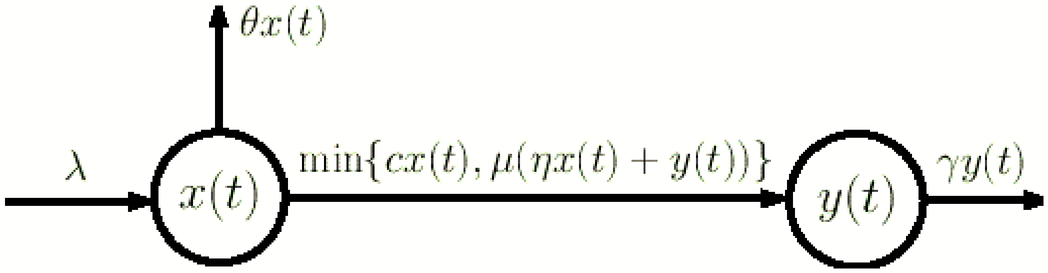
Piece selection

- Initially, a few random pieces
  - anything is better than nothing
  - easy to find at the beginning
- Then, rarest-first in neighborhood
  - become less dependent on seed
  - more interested by peers
- Finally, “end game” mode
  - look for missing pieces aggressively
  - send requests to all peers
  - cancel requests after last pieces are collected

BitTorrent performance

- Modeling and analysis
  - [QS04] Dongyu Qiu, R. Srikant. Modeling and Performance Analysis of Bit Torrent-Like Peer-to-Peer Networks. SIGCOMM 2004 [BitTorrent]

Fluid model



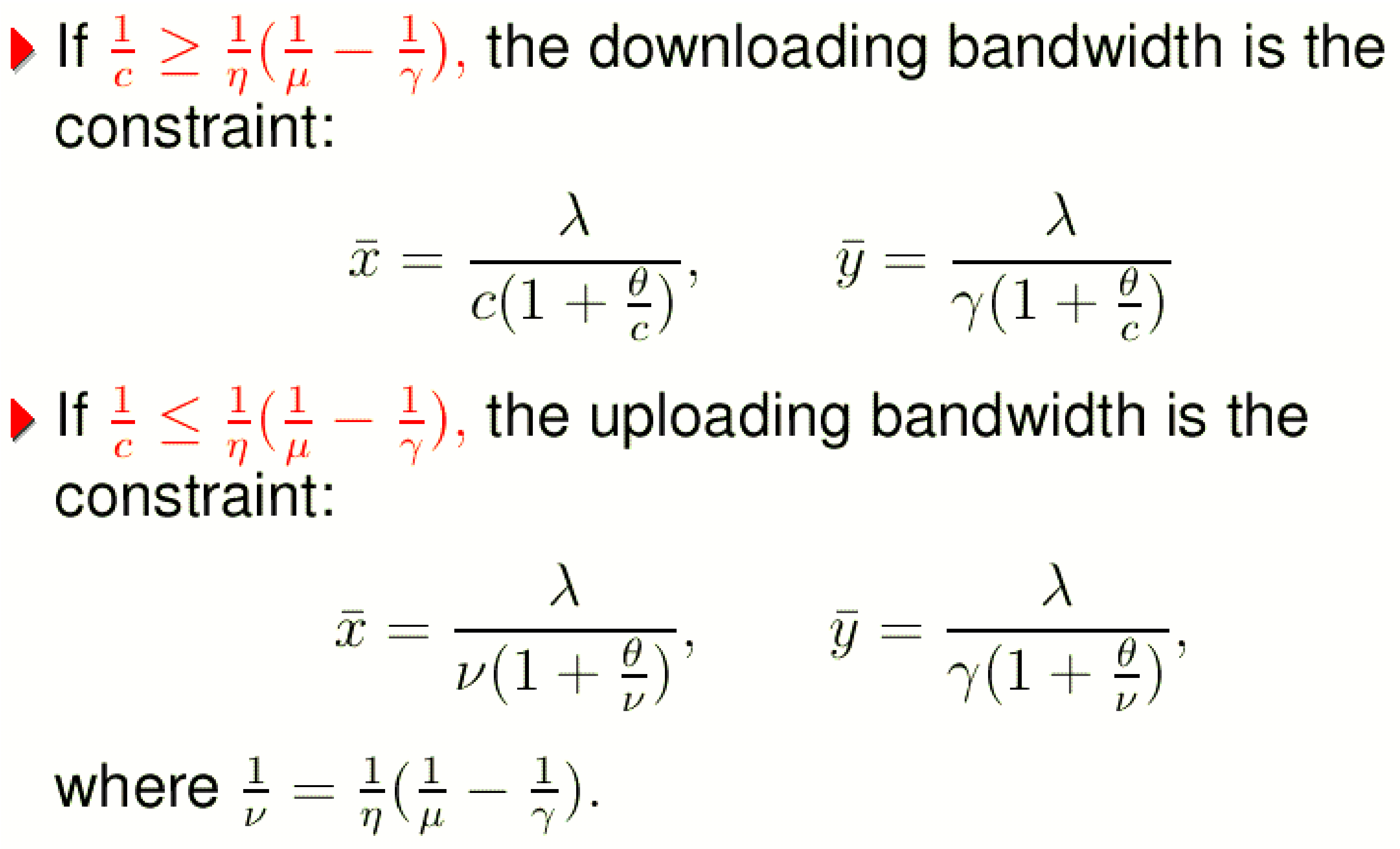
- $x(t)$: number of downloaders, $y(t)$: number of seeds
- λ: arrival rate of new requests
- θ: the rate at which a downloader aborts the download
- μ: uploading bandwidth of a peer
- η: effectiveness parameter (Yang and de Veciana)
- c : downloading bandwidth of a peer
- γ: seed departure rate

$$\frac{\mathrm{d}x}{\mathrm{d}t} = \lambda - \theta x(t) - \min\{cx(t), \mu(\eta x(t) + y(t))\}$$

$$\frac{\mathrm{d}y}{\mathrm{d}t} = \min\{cx(t), \mu(\eta x(t) + y(t))\} - \gamma y(t)$$

Steady-state performance

- dx(t)/dt=dy(t)/dt=0



- If $\frac{1}{c} \geq \frac{1}{\eta}(\frac{1}{\mu} - \frac{1}{\gamma})$, the downloading bandwidth is the constraint:

$$\bar{x} = \frac{\lambda}{c(1+\frac{\theta}{c})}, \qquad \bar{y} = \frac{\lambda}{\gamma(1+\frac{\theta}{c})}$$

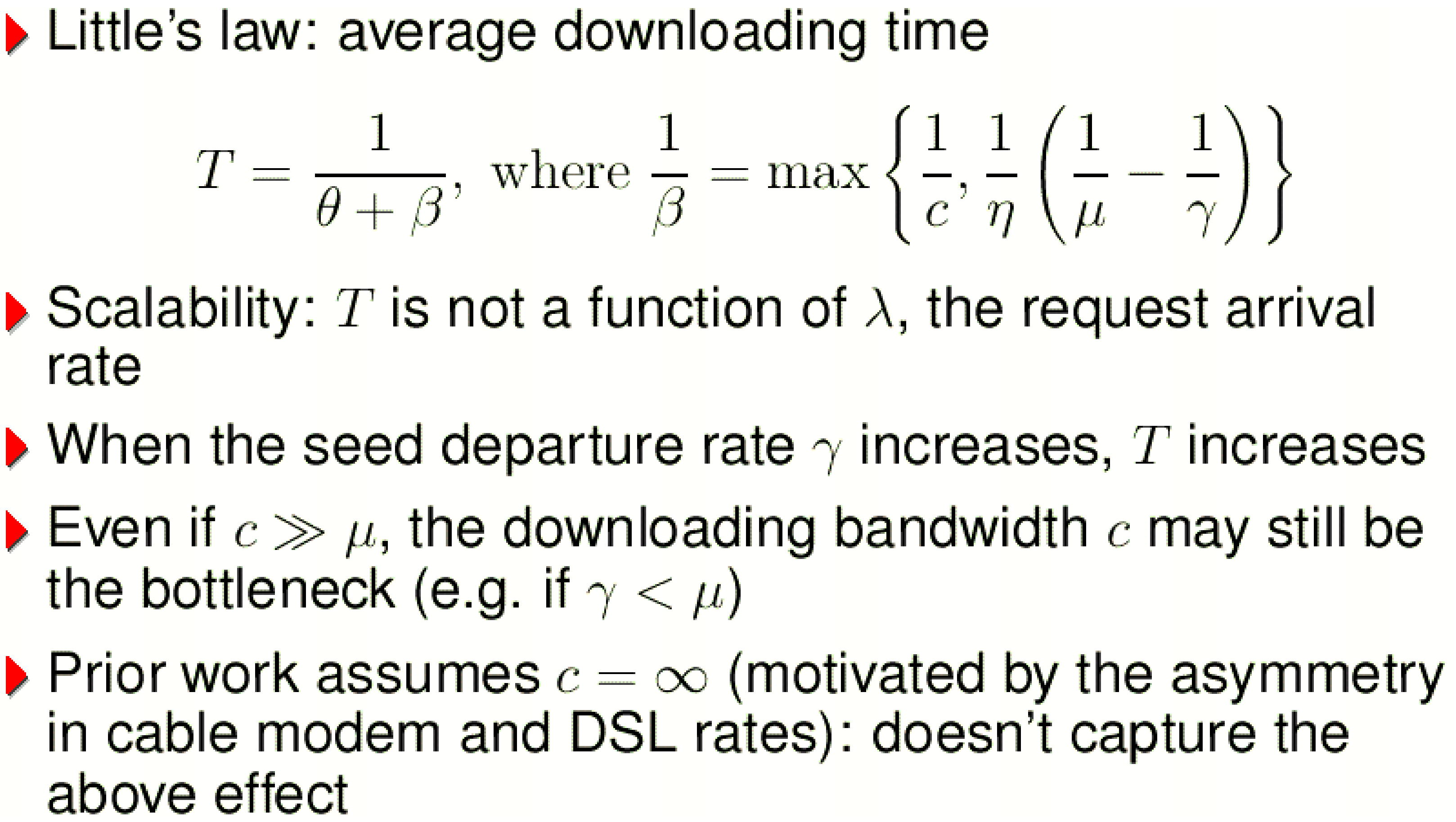
- If $\frac{1}{c} \leq \frac{1}{\eta}(\frac{1}{\mu} - \frac{1}{\gamma})$, the uploading bandwidth is the constraint:

$$\bar{x} = \frac{\lambda}{\nu(1+\frac{\theta}{\nu})}, \qquad \bar{y} = \frac{\lambda}{\gamma(1+\frac{\theta}{\nu})},$$

where $\frac{1}{\nu} = \frac{1}{\eta}(\frac{1}{\mu} - \frac{1}{\gamma})$.

Analytical insights

- Intrinsic scalability



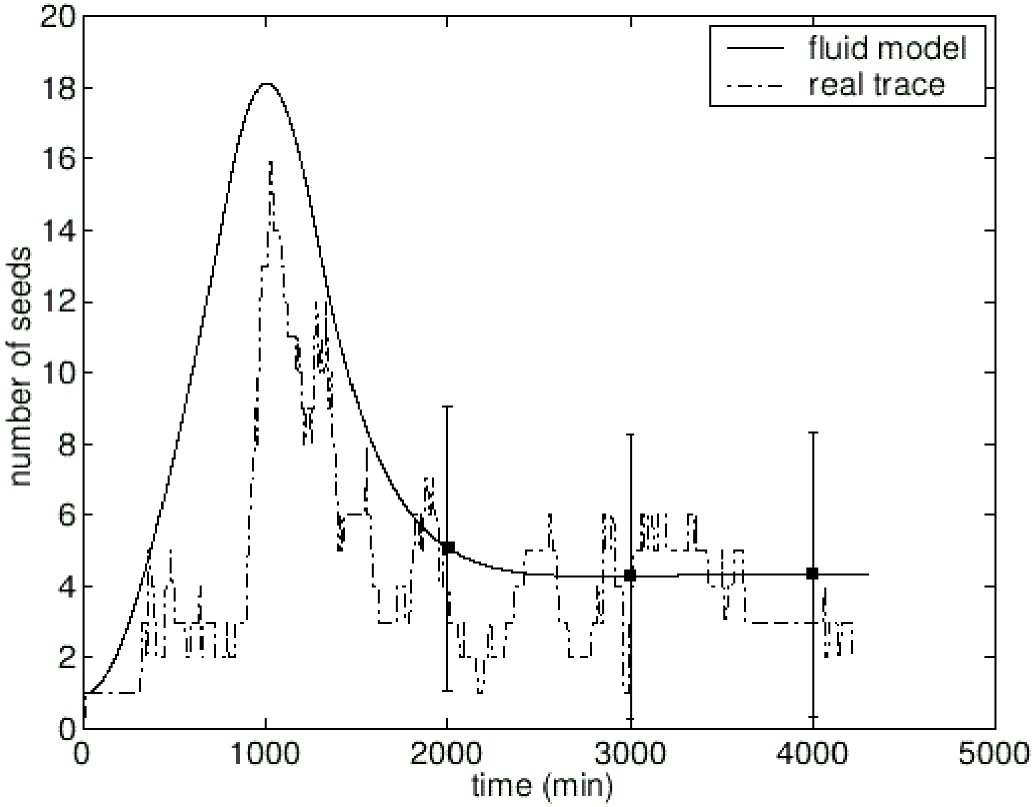
- Little's law: average downloading time

$$T = \frac{1}{\theta + \beta}, \text{ where } \frac{1}{\beta} = \max\left\{\frac{1}{c}, \frac{1}{\eta}\left(\frac{1}{\mu} - \frac{1}{\gamma}\right)\right\}$$

- Scalability: T is not a function of λ, the request arrival rate
- When the seed departure rate γ increases, T increases
- Even if $c \gg \mu$, the downloading bandwidth c may still be the bottleneck (e.g. if $\gamma < \mu$)
- Prior work assumes $c = \infty$ (motivated by the asymmetry in cable modem and DSL rates): doesn't capture the above effect

Evaluation results

- Fluid model vs real trace
  - Ӈ the number of sees in the system



More discussion

This lecture

- BitTorrent
  - P2P swarming
  - protocol overview
  - performance analysis
- Explore further
  - measurement-based modeling
  - measurement-based performance analysis
  - BitTorrent extensions
    - http://wiki.theory.org/BitTorrentSpecification

Next lecture

- Next: Skype
  - [BS06] Salman A. Baset and Henning Schulzrinne, "An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol", IEEE Infocom 2006. [Skype]

- Notice
  - reading list and schedule are online
  - check the list and submit reading summary on time

* connext->wiki->reading groups and schedule; papers announced a week before due

Reading summaries

- Challenge the paper presented by the other group
  - The problem(s)
  - Main Idea(s)
  - Major (at least three) strengths
  - ***Major (at least three) Weaknesses, then and now***
  - Possible improvement
- Template
  - http://www.cs.uvic.ca/~pan/csc466/rs.txt (Text)
  - http://www.cs.uvic.ca/~pan/csc466/rs.tex (LaTeX)

more detailed reading guideline: http://www.cs.uvic.ca/~pan/csc466/reading.txt

Midterm on June 15, 2015

- Materials
  - lectures and in-class Q&A and discussions
  - papers discussed during the lectures
  - connex->lectures->required reading (10 papers)
  - three extra papers: [Pastry], [Kademlia], [Tapestry]
- Open papers/notes exam
  - a computer without Internet connection is allowed
  - i.e., cannot get help outside the classroom
  - neither from other students in the classroom