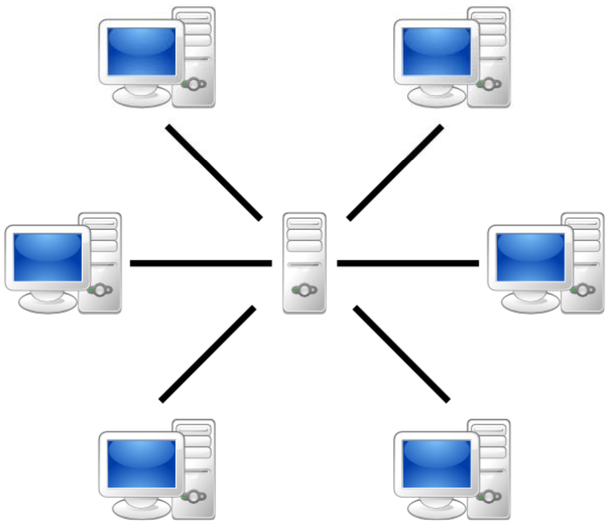
Beyond File Sharing: Recent Technologies and Trends in Peer-To-Peer Systems

Minghua Chen[1] and Sudipta Sengupta[2]
[1]The Chinese University of Hong Kong
[2]Microsoft Research at Redmond

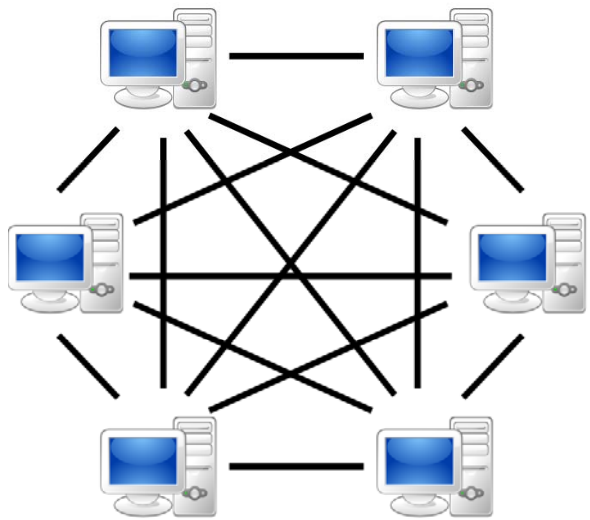
Outline

- P2P and its history
- P2P modeling
- Streaming capacity of P2P systems
- Delay minimization of P2P systems
- P2P Video-on-Demand (VoD) Systems
- ISP Friendliness in P2P
- Utility maximization in P2P systems and its application to P2P conferencing
- Queuing models for P2P systems
- Network coding in P2P systems

P2P: Scalable Content Distribution Infrastructure



Server-client



Peer-to-peer

(pictures from wikipedia)

A Brief History of P2P

- Napster [Shawn Fanning, 1999 ~ 2001]
- Gnutella [Justin Frankel and Tom Pepper, 2000 ~]
- BitTorrent [Bram Cohen, 2001 ~]
- CoolStreaming [Xinyan Zhang (CUHK), 2004 ~]
  - PPLive, UUSee, PPStream, Anysee, Thunder
  - Octoshape, Hulu, Dyyno (by Bernd Girod)…
- P2P storage systems are emerging, e.g., Wuala [2006~]
- P2P VoD [PPLive, UUSee, PPStream 2006~]
- P2P conferencing [Chen et al. 2008~]

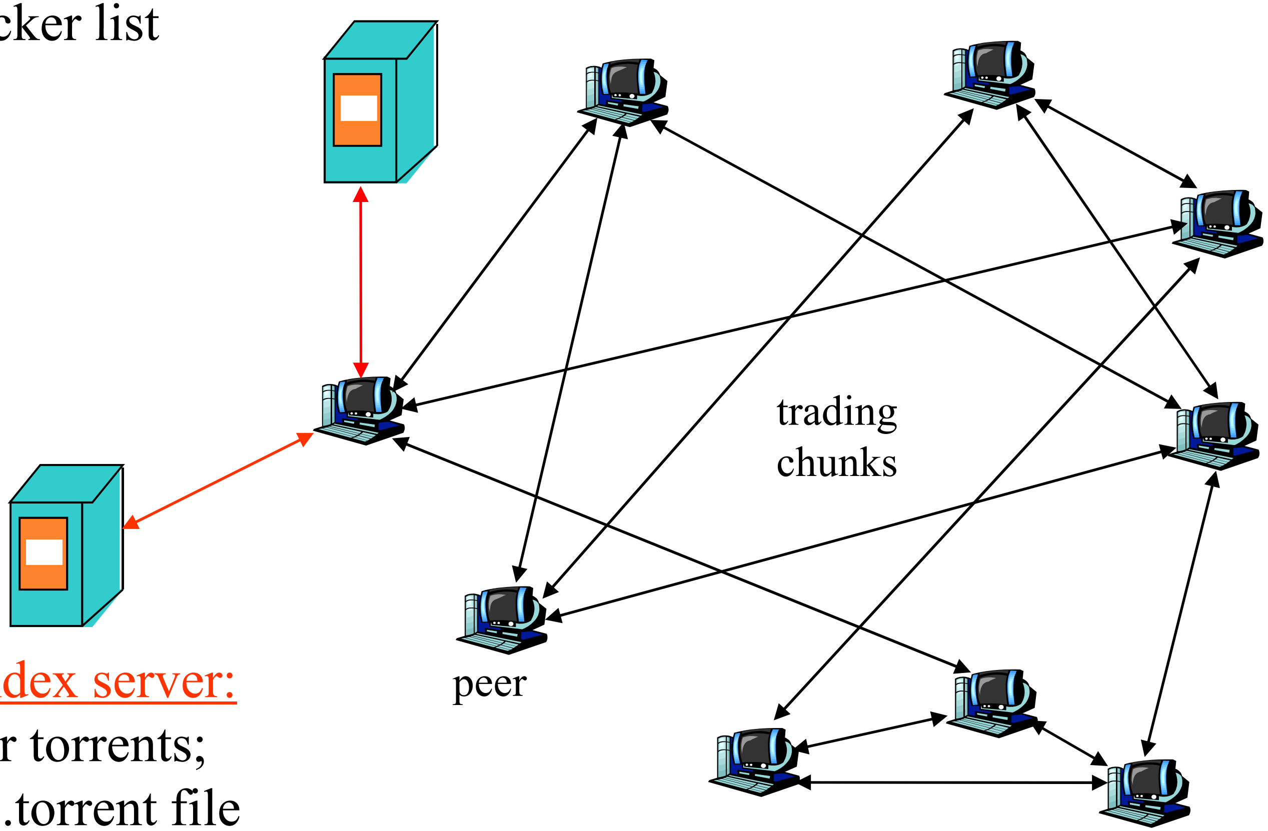
BitTorrent

- A Peer-to-Peer Content Distribution Protocol/ Program
- Developed by Bram Cohen in 2001
  - Bram grew up in upper west side of Manhattan, NYC
- First version written in Python

BitTorrent

tracker: tracks peers in torrent; provides tracker list

torrent: group of peers exchanging chunks of a file



torrent index server: search for torrents; provides .torrent file

BitTorrent – Terminology (1)

- File divided into **pieces**
  - 1 piece =16 blocks = 256 KB
- Seeds and leechers
  - Seed has **complete** file. **Upload only**
  - Leecher has **incomplete** file. **Upload/download**
- Buffer Map
  - Peers advertise pieces they have to neighbors

BitTorrent – Terminology (2)

- Regular Unchoke -- **Tit-for-Tat**
  - Peer sends blocks to **n-1** neighbors currently sending it data *at* **highest rate** (n is # of upload slots)
- Optimistic Unchoke
  - Peer also sends blocks to one **random** neighbor
- Each file has unique **infohash**
  - Hash of concatenation of hashes of all pieces

BitTorrent Ecosystem

- Open protocol
  - 50+ client implementations
  - Dozens of tracker implementations
  - Dozens of torrent location sites
- 5 million simultaneous users & growing
- Evolving:
  - Peer discovery: DHTs, gossiping
  - Proprietary protocols, private torrents

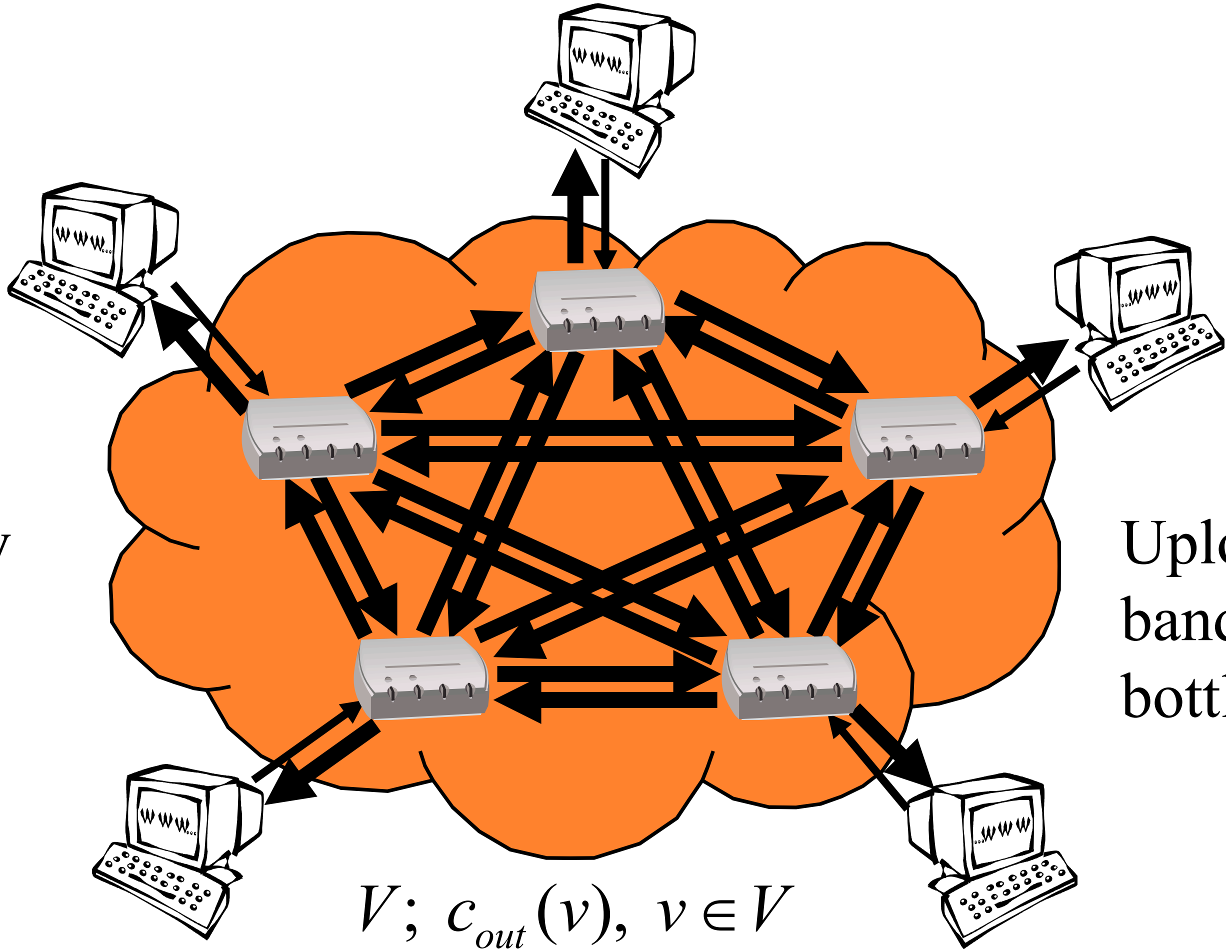
Beyond BitTorrent

- A vibrant research and fast industrializing area
  - Systems: streaming, VoD, conferencing, storage
  - QoS of static systems: throughput, delay
  - QoS of dynamic systems: stability and delay performance
  - ISP-friendliness
  - Network coding aided P2P systems
  - Incentive
  - Security

I. Modeling P2P Systems

P2P Networks

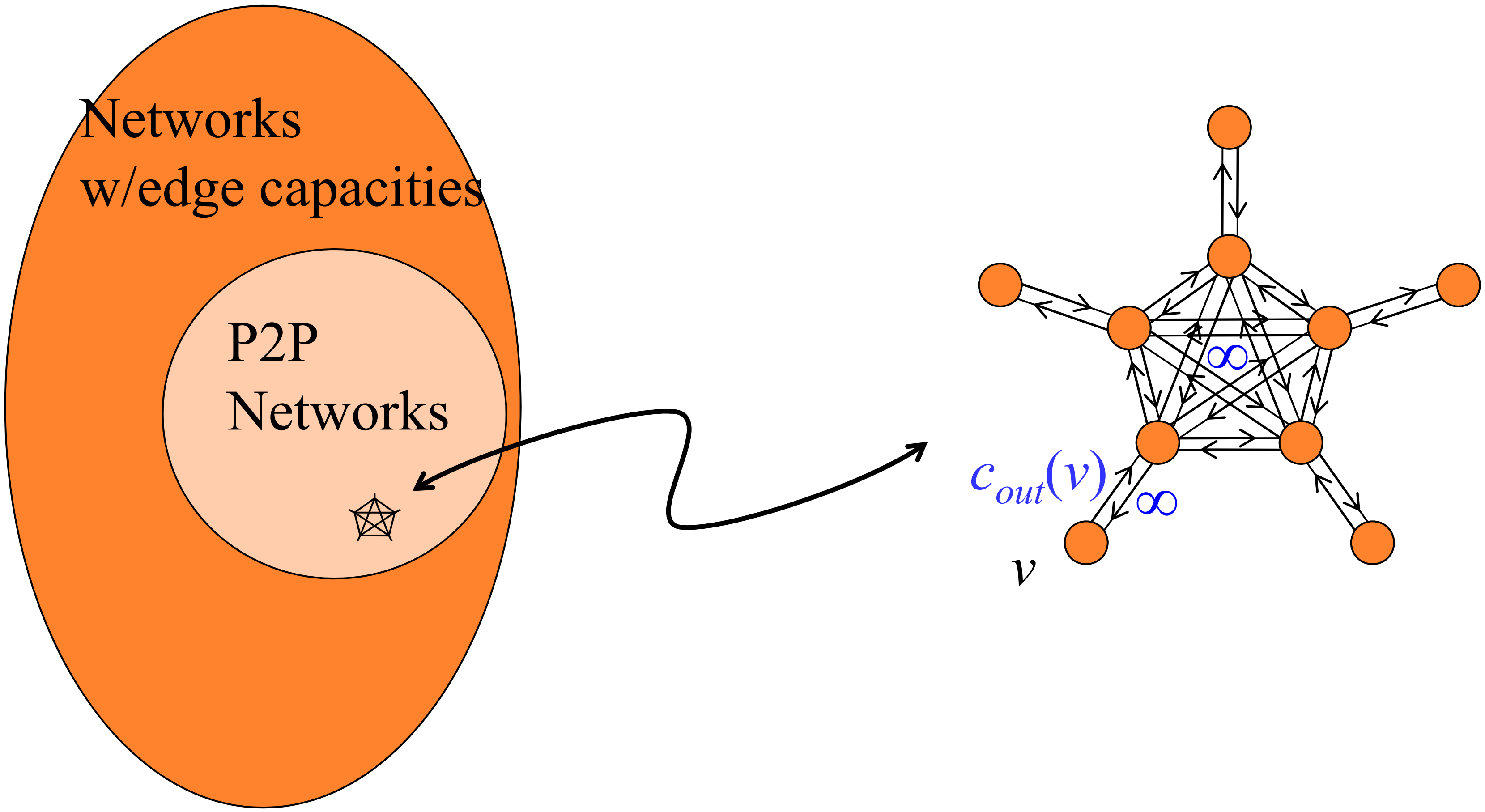
Completely connected overlay



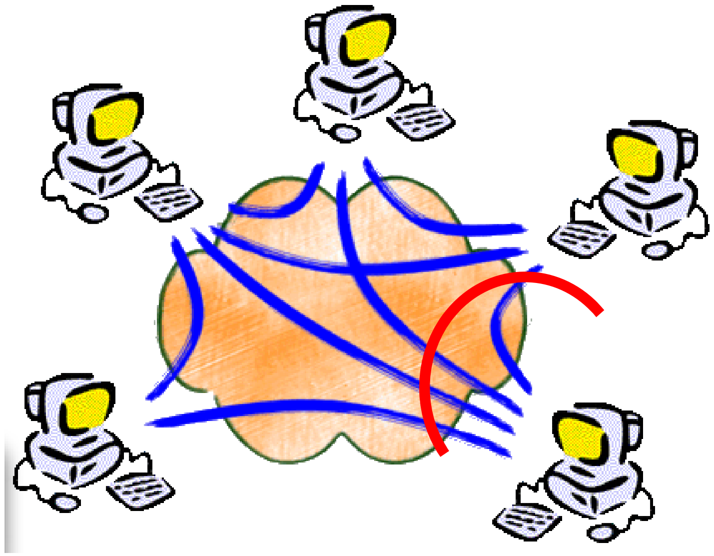
Upload bandwidth is bottleneck

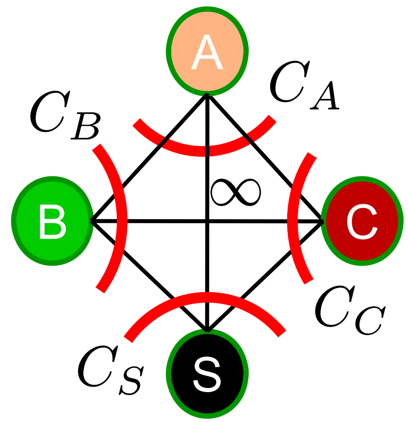
$V;\ c_{out}(v),\ v \in V$

P2P Network as Special Case



Modeling P2P Overlay Networks





- Overlay networks are node-capacity constrained
  - A “link”: a TCP/UDP connection
  - Node uplinks are the capacity bottleneck
  - Total out-going link rate $\leq$ uplink capacity

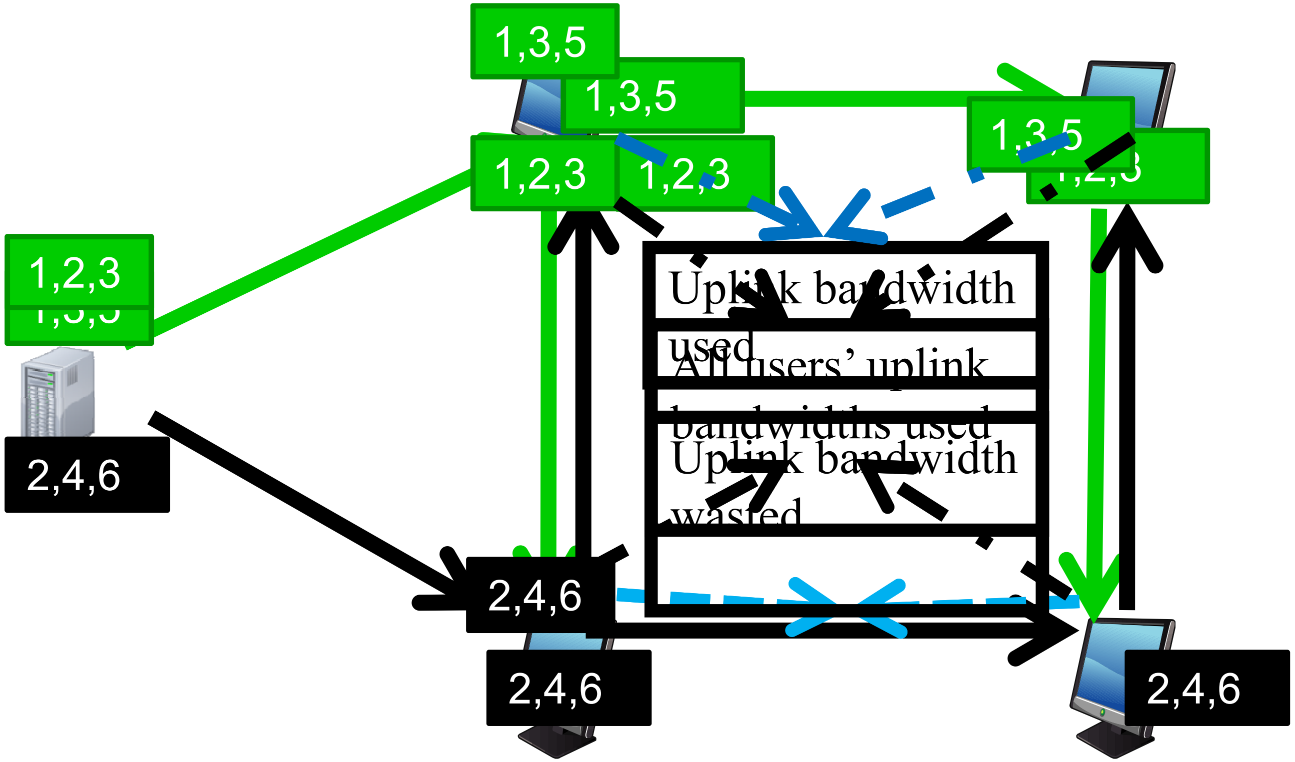
II. QoS in Static Peer-to-Peer Systems
A. Streaming Capacity

P2P Streaming Systems Are Popular Today

- High quality (700+kbps) streaming of Beijing Olympic in the summer of 2008 by PPLive, UUSEE, etc.
- Single-rate streaming

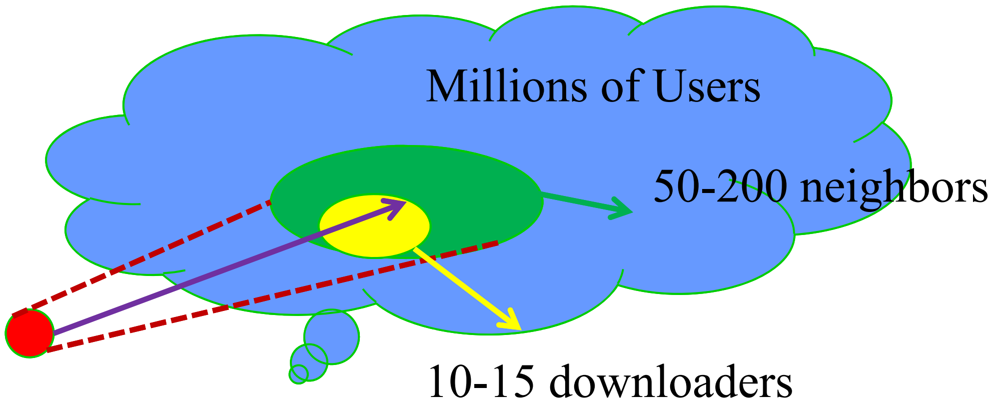
Tree-based Streaming: Multiple Trees

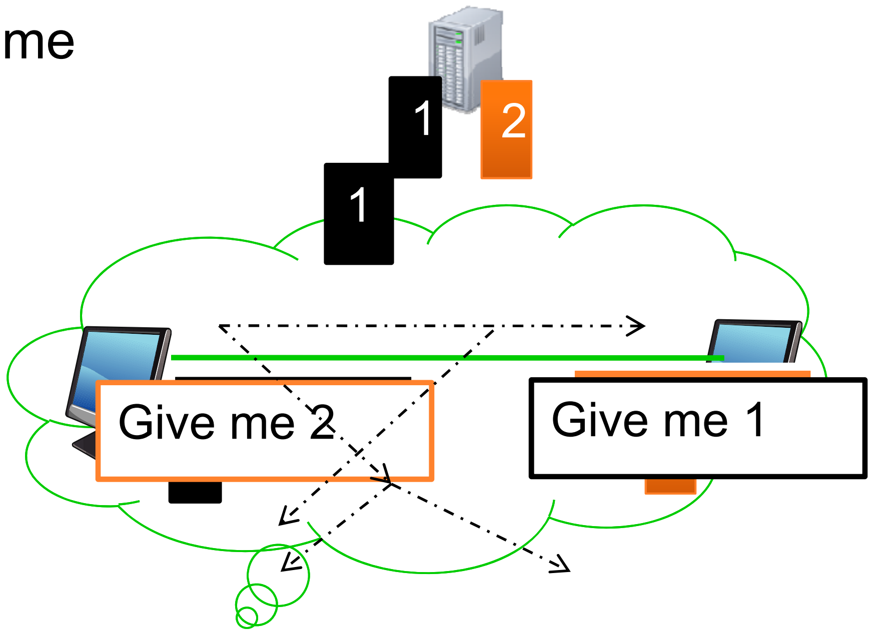
- Multiple trees (multi-tree) approach: high efficiency



Commercial P2P Streaming Systems

- PPLive and UUSee [Wu-Li 07, Hei-Liang-Liang-Liu-Ross 06]
  - 10k+ channels reported in UUSEE (each channel >400kbps)
  - 15K users per channel in the peak time
  - >1 Million users online in peak time





- Still evolving: hybrid P2P+CDN: SmoothHD

Fundamental Questions

- What is the *streaming capacity* of P2P streaming systems?

Streaming capacity = maximum achievable rate for all receivers.

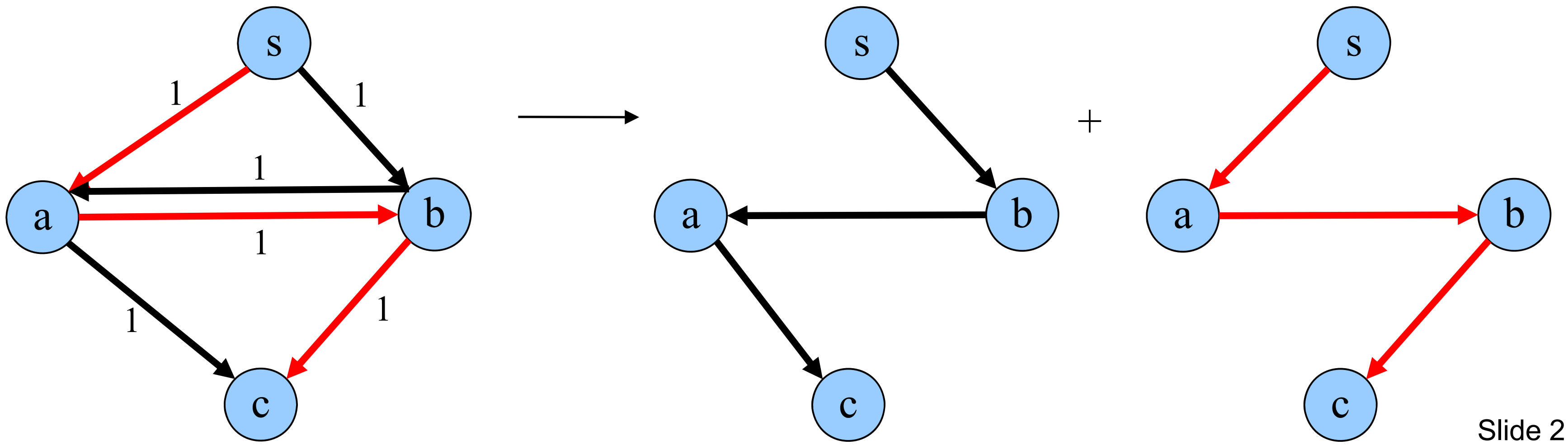
- How to achieve the limit?

Outline

- Peer-to-peer (P2P) and its history
- P2P modeling and streaming capacity
  - Modeling P2P overlay networks
  - Streaming capacity for the full-mesh case
  - Streaming capacity for general cases
- Summary

Story for Underlay Networks

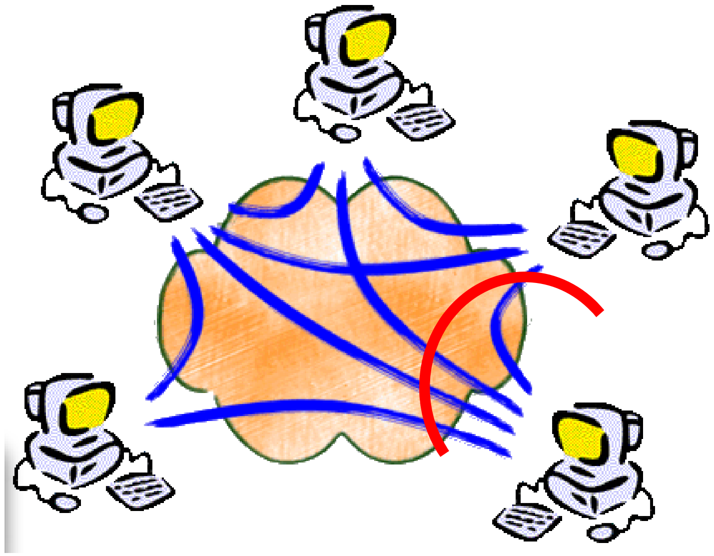
- Underlay networks are link-capacity constrained
  - A "link": a physical fiber/DSL links
  - Directed link are the capacity bottleneck
- [Edmond 72] Packing polynomial number of spanning trees obtains maximum broadcast rate

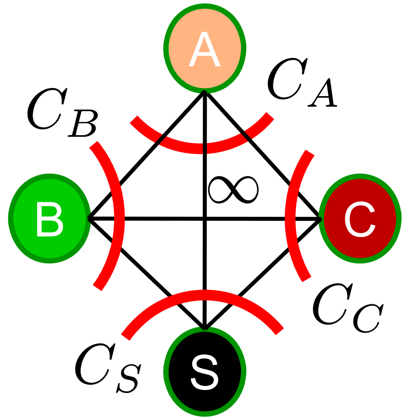


Story for Underlay Networks

- Underlay networks are link-capacity constrained
  - A “link”: a physical fiber/DSL links
  - Directed link are the capacity bottleneck
- [Edmond 72] Packing polynomial number of spanning trees obtains maximum broadcast rate
- [Jain 03] Maximizing multicast rate by packing Steiner trees is NP-hard
- Maximizing multicast rate by Network Coding is polynomial-time solvable (a long list of references)

Modeling P2P Overlay Networks

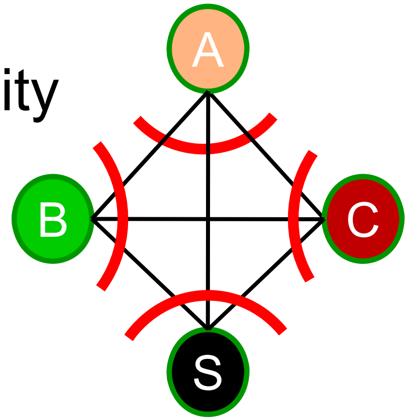




- Overlay networks are node-capacity constrained
  - A "link": a TCP/UDP connection
  - Node uplinks are the capacity bottleneck
  - Total out-going link rate $\leq$ uplink capacity

Full-mesh With Upload Constraints

- Fully connected graph
- Total out-going link rate $\leq$ uplink capacity



- Server: S
- N heterogeneous receivers: V – {S}
- Streaming rate: r

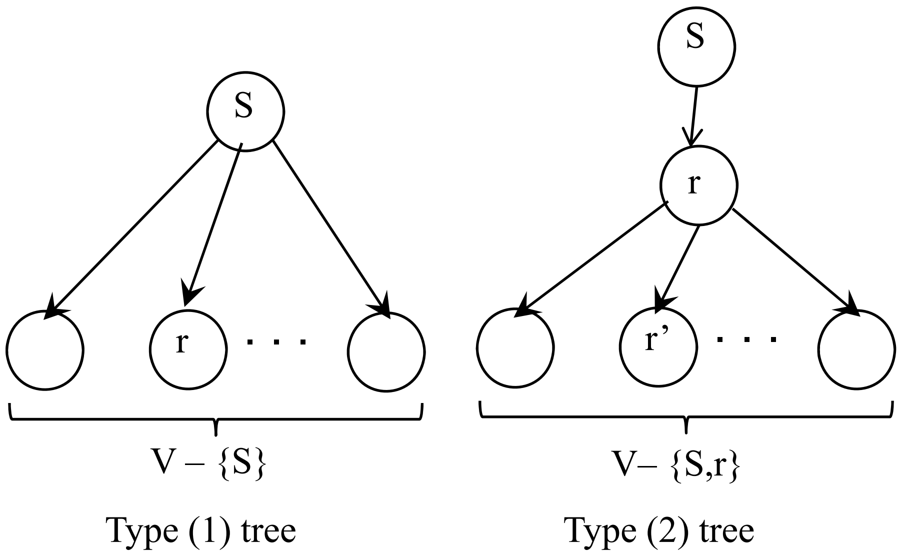
- r satisfies r $\leq$ C_S, and cut-set bound

$$\underbrace{Nr}_{\text{total receiver demand}} \leq \underbrace{C_S + \sum_{v \in V-\{S\}} C_v}_{\text{total possible system supply}}$$

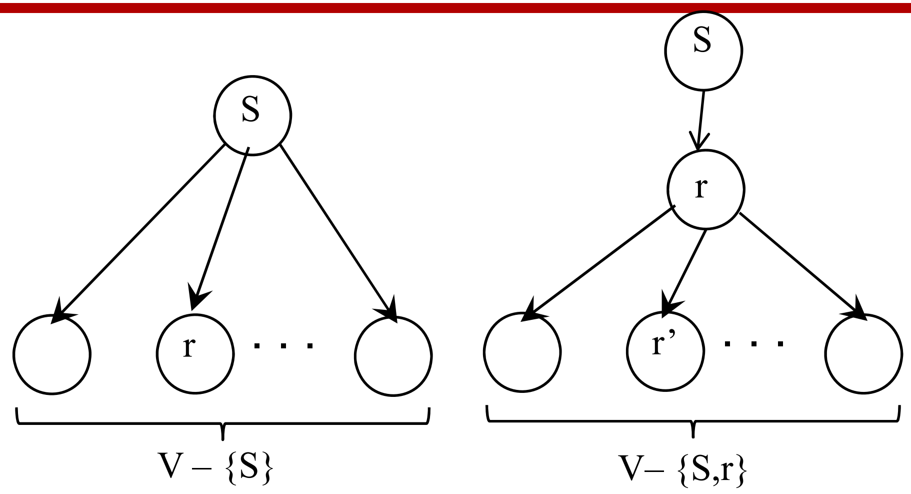
Full-mesh With Upload Constraints

$$r \leq \frac{1}{N}\left(C_S + \sum_{v \in V-\{S\}} C_v\right) \rightarrow \mu \triangleq \frac{1}{N}\sum_{v \in V-\{S\}} C_v$$

- To achieve the bound
  - Maximize total system supply
  - Maximize efficiency (every transmission is useful)



Full-mesh With Upload Constraints

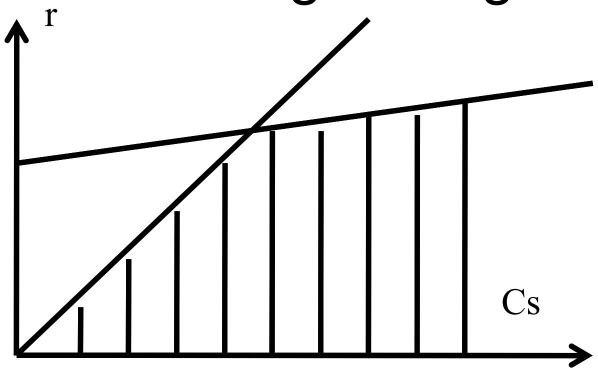


Type (1) tree

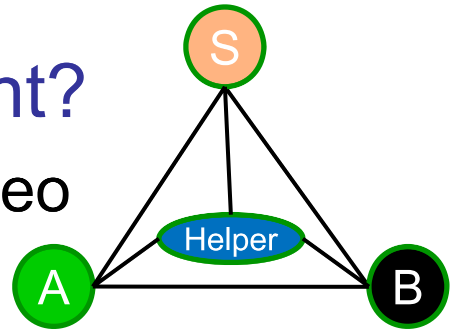
Type (2) tree

- Therefore, the streaming capacity is given by [Li-Chou-Zhang 05, Mundinger-Weber-Weiss 05, Chiu-Yeung-Huang-Fan 06, Kumar-Liu-Ross 07]

$$\boxed{\min\left(C_S, \frac{1}{N}\left(C_S + \sum_{v \in V-\{S\}} C_v\right)\right)}$$



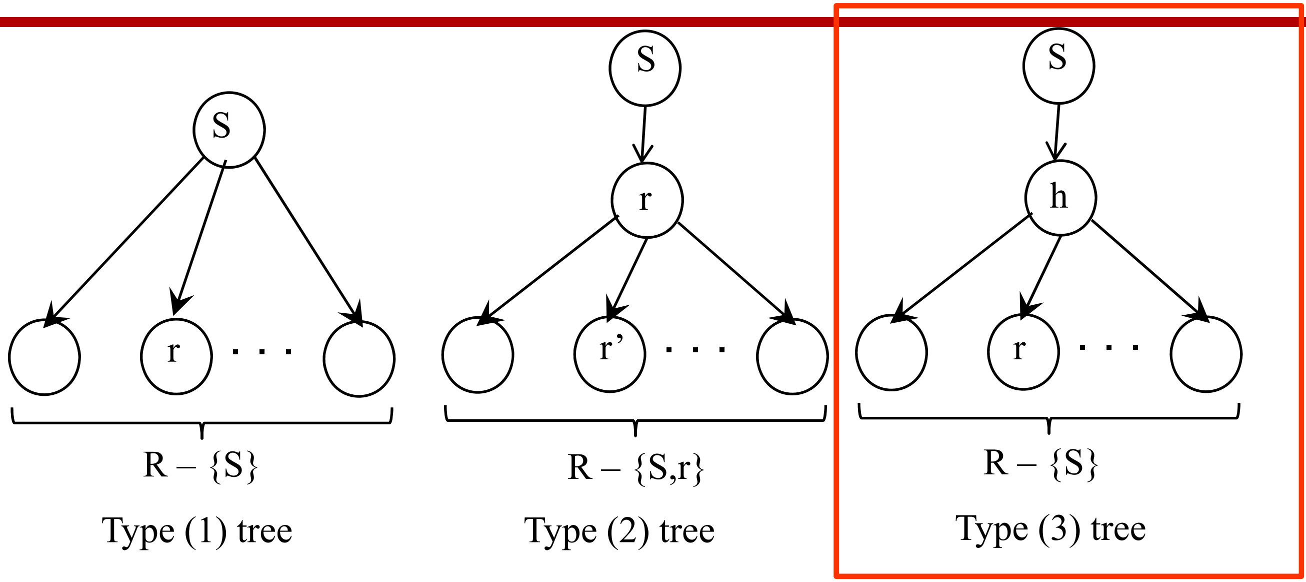
Full-mesh With Upload Constraints

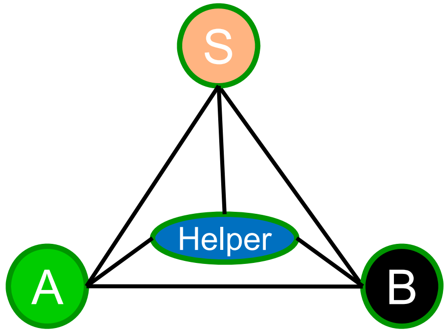


- What if helpers (Steiner nodes) present?
  - Helpers not interested in watching the video
  - Just there to help
  - Can be Akamai servers

- Same insights still apply
  - Maximize total system supply
  - Maximize efficiency

Full-mesh With Upload Constraints





- Streaming capacity (with helper presence) is achieved by packing MutualCast Tree [Li-Chou-Zhang 05, Chen-Ponec-Sengupta-Li-Chou 08]

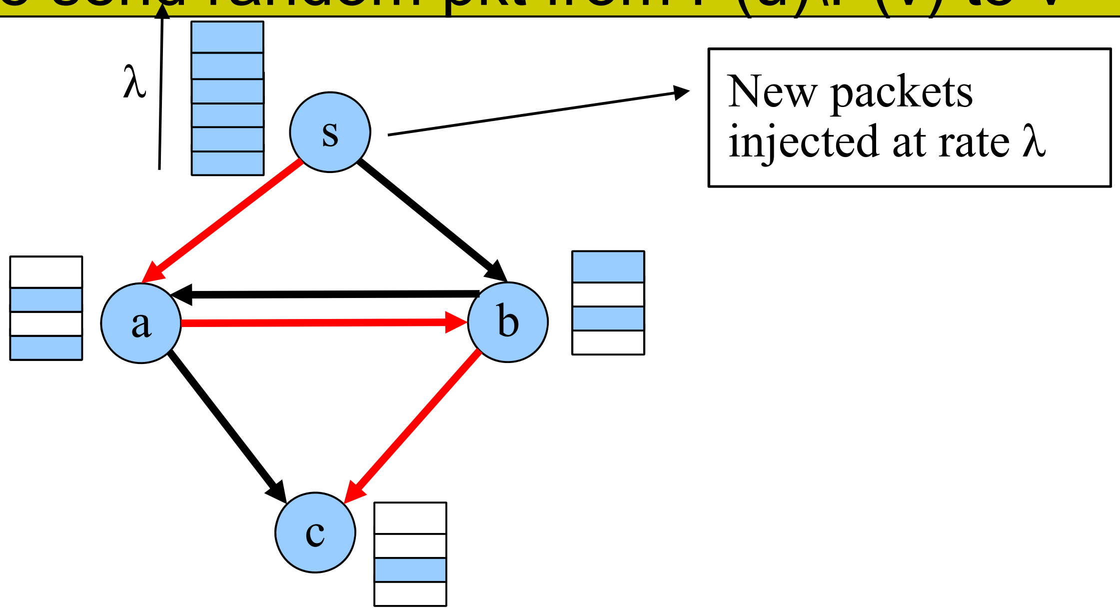
$$\boxed{\min\left(C_S, \frac{1}{N}\left(C_S + \sum_{v\in R-\{S\}} C_v + \underbrace{\left(1-\frac{1}{N}\right)\sum_{h\in H} C_h}_{\text{supply from helpers}}\right)\right)}$$

Mesh-based Solution
[Twigg-Massoulié- Gkantsidis- Rodriguez 07]

- Let P(u) = packets received by u

for each node u

- choose a neighbour v maximizing |P(u)\P(v)|
- If u=source, and has fresh pkt, send random fresh pkt to v
- Otherwise send random pkt from P(u)\P(v) to v



RU packet forwarding: Main result

Assumptions:

- G: arbitrary edge-capacitated graph
- Min(mincut(G)): λ^*
- Poisson packet arrivals at source at rate $\lambda < \lambda^*$
- Pkt transfer time along edge (u,v): Exponential random variable with mean 1/c(u,v)

Theorem

With RU packet forwarding,

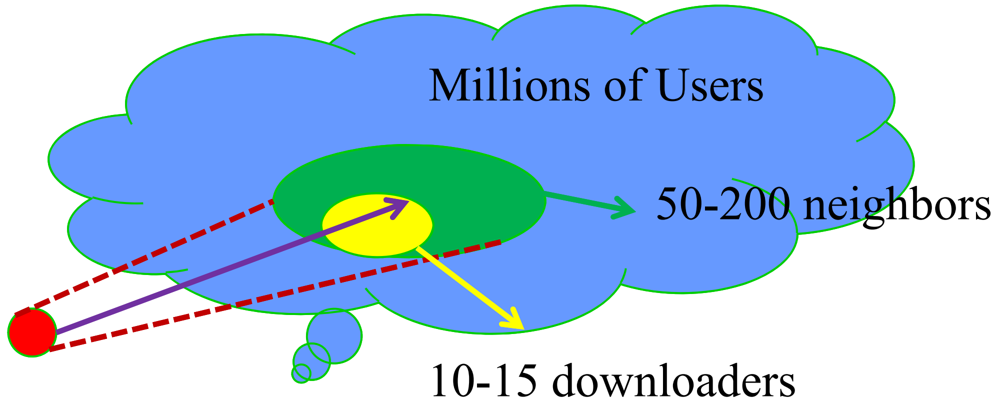
Nb of pkts present at source not yet broadcast:

A stable, ergodic process.

Design for broadcast scenarios. Optimal if the graph is full-mesh.

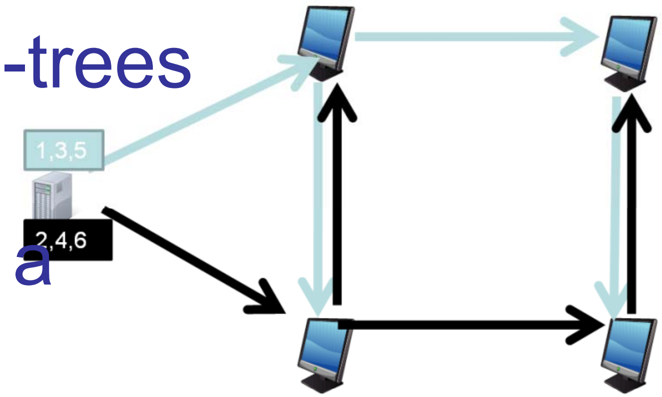
So Far It Is Cool, But…

- Full-mesh requires every peer connects to every other peer!
  - Connection overhead drains out peer's resource
- For large commercial streaming systems, the graph is non-full-mesh, and is given



General Networks With Upload Constraints

- P2P streaming = packing multi-trees on overlay graph
- Streaming capacity problem is a multi-tree max-flow problem
  - Number of tree rate variables: exponential (NP-hard, Sudipta-Liu-Chen-Chiang-Li-Chou 08)



Streaming rate

Tree rate

$$\text{maximize} \quad r = \sum_{t \in T} y_t \qquad (1)$$

$$\text{subject to} \quad \sum_{t \in T} m_{v,t} y_t \le C_v \, , \ \forall v \in V \qquad (2)$$

tree degree

$$y_t \ge 0 \, , \ \forall t \in T \qquad (3)$$

uplink constraint

$$\text{variables} \quad y_t, \forall t \in T \qquad (4)$$

General Networks With Upload Constraints

- SC problem is hard:
  - ***Exp-number of variables***
  - Linear number of constraints

Streaming rate

Tree rate

$$
\begin{array}{ll}
\text{maximize} & r = \sum_{t \in T} y_t \\
\text{subject to} & \sum_{t \in T} m_{v,t} y_t \le C(v),\ \forall\, v \in V \\
 & y_t \ge 0\ \forall\, t \in T \\
\text{variables} & y_t, \forall t \in T
\end{array}
$$

- Dual Problem is also hard:
  - Price p(v) for each v
  - Linear number of variables
  - ***Exp-num of constraints***

Node price: price for each uplink

$$
\begin{array}{ll}
\text{minimize} & \sum_{v \in V} C(v) p_v \\
\text{subject to} & \sum_{v \in V} m_{v,t} p_v \ge 1,\ \forall\, t \in T, \\
 & p_v \ge 0\ \forall\, v \in V \\
\text{variables} & p_v, \forall v \in V
\end{array}
$$

Tree price

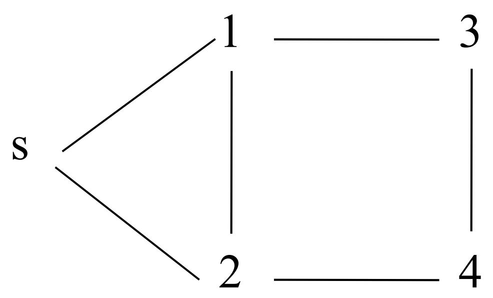
Make It Easy? Solve Two Problems Jointly!

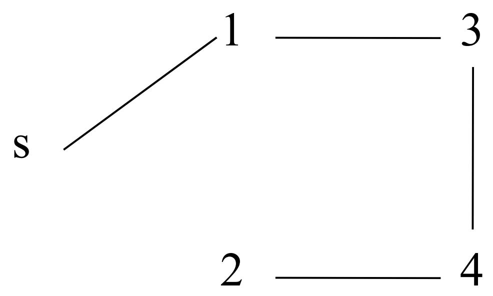
- Solving the problem approximately
  - Primal-dual technique modified from Garg & Konemann [Garg-Konemann 98]
- Basic observations
  - Solving the problem optimally may require packing exponential number of trees
  - Solving the problem approximately requires only a set of polynomial number of trees

Iterative Algo. to Find Streaming Capacity

- Outer loop
  - Inner loop
    - Solve Smallest Price Tree (SPT) problem
    - Record the “good” tree found
  - Update price of each nodes
  - Terminate when we have enough “good” trees

Smallest Price Tree (SPT) Problem



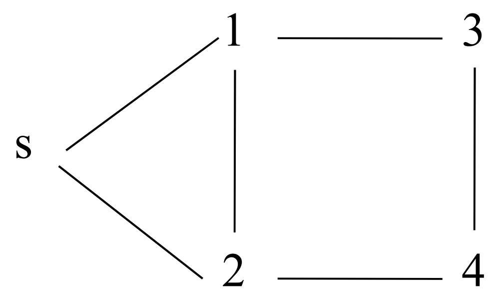


- Given a graph G=(V, E) and prices for traversing each node

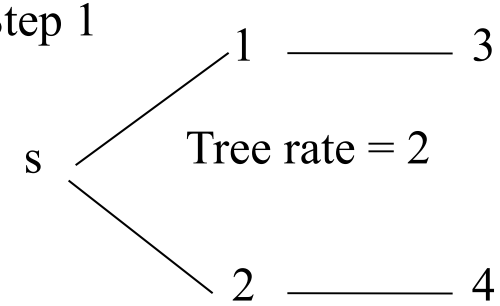
- Find a tree with smallest price, connecting server S and all N receivers
  - N=1: shortest path problem (poly. time solvable)
  - N>1: NP-complete in general

Example

- C[s,1,2,3,4]=[5,4,2,6,3]
- ϵ = 0.1, p(v)=0.1



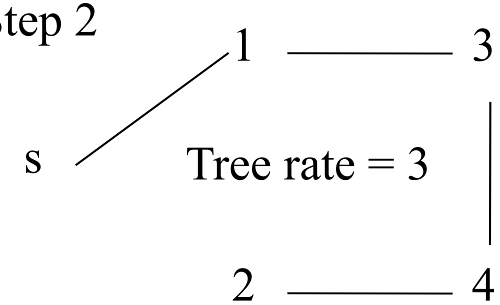
Step 1



$$p(s)=\delta(1+\varepsilon\frac{2*2}{5})=0.108 \qquad p(1)=\delta(1+\varepsilon\frac{1*2}{4})=0.105$$

$$p(2)=\delta(1+\varepsilon\frac{1*2}{2})=0.11 \qquad p(3)=p(4)=\delta=0.1$$

Step 2



$$p(s)=0.108*(1+\varepsilon\frac{1*3}{5})=0.11448 \qquad p(1)=0.105*(1+\varepsilon\frac{1*3}{4})=0.11287$$

$$p(2)=0.11 \qquad p(3)=0.1*(1+\varepsilon\frac{1*3}{6})=0.105 \qquad p(4)=0.1*(1+\varepsilon\frac{1*3}{3})=0.11$$

SPT Tree Finding (Challenging Part)

	Full-mesh graph		General graph	
	No Helper	W/ Helpers	No Helper	W/ Helpers
No tree degree bound	Spanning tree Poly solvable	Steiner tree Poly solvable	Spanning tree Poly solvable	Steiner tree NP-hard →Group Steiner tree → 1/log(N)
Tree degree bound	Spanning tree Poly solvable	Steiner tree Poly solvable	Spanning tree NP-hard ¼ approx.	Steiner tree NP-hard open

See Sengupta-Liu-Chen-Chiang-Li-Chou 08 for references.

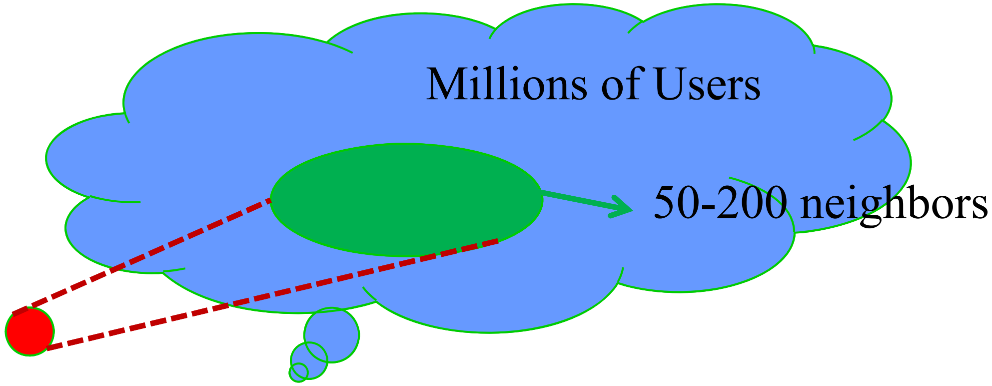
Optimality and Time Complexity

- If SPT finding is polynomial-time solvable
  - Then achieve $(1 - \epsilon)$ * streaming capacity
- If SPT finding is NP-hard, and exists θ–approximation algorithm (θ <1)
  - Then achieve $(\theta - \epsilon)$* streaming capacity
- Time complexity
  - The iterative algorithm takes O(N log(N)) rounds

Big Picture

- Full mesh graph: Packing MutualCast trees
- General graph: Garg-Konemann framework approaches optimality (a centralized solution)
  - Distributed algorithms for special case: Mossouli et al. 07, a modified version of Ho and Viswanathan 07
- One more degree of freedom to explore: optimizing the graph (by neighbor selection) to further improve streaming capacity!

Joint Neighbor Seletion And Rate Optimization

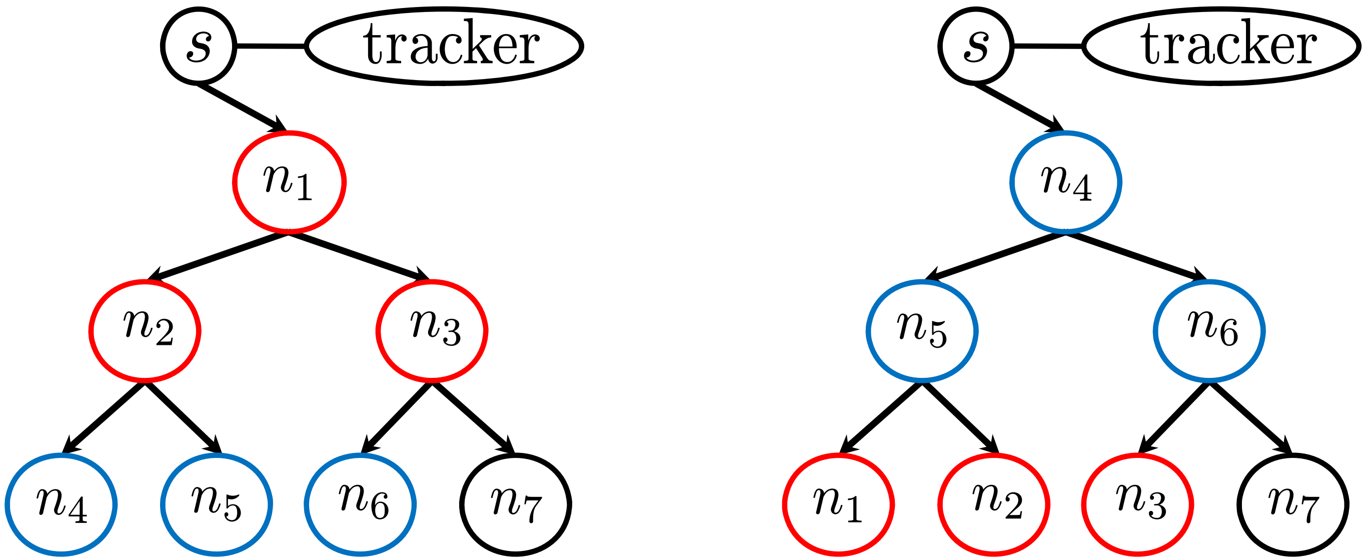


- Choose a sub-graph satisfying node degree bound
  - Each peer has at most M neighbors
  - Bounded overhead in maintaining TCP/UDP connections
- Over the subgraph, optimize the streaming rate

- This joint problem is NP-hard in general [Liu-Chen-Sengupta-Chiang-Li-Chou 10]

Simple Case: Homogeneous Peers

- One server, 8 homogeneous peers, unit capacity
- Packing interior-node-disjoint trees achieve streaming rate 1
  - (CoopNet) Padmanabhanet al. 02, (SplitStream) Castro et al. 03,



How about Heterogeneous Peers?

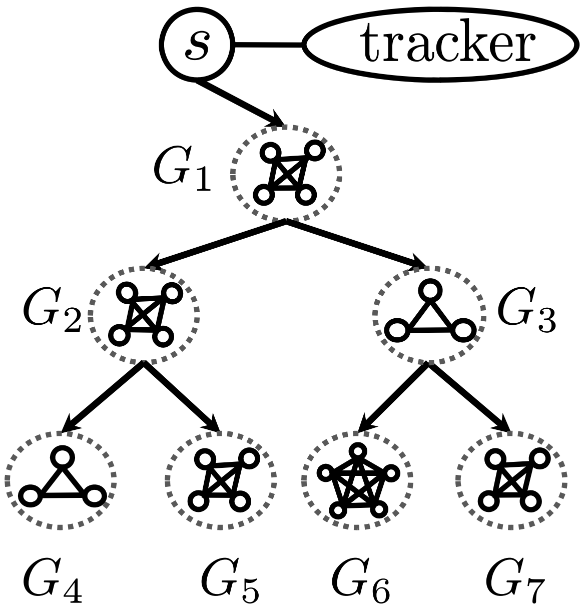
- (centralized) Bubble algorithm [Liu-Chen-Sengupta-Chiang-Li-Chou 10]: packing degree bounded trees
- Key insights:
  - Nodes with large capacity on top of the trees
  - Carefully swap exhausted intermediate nodes with leaf nodes

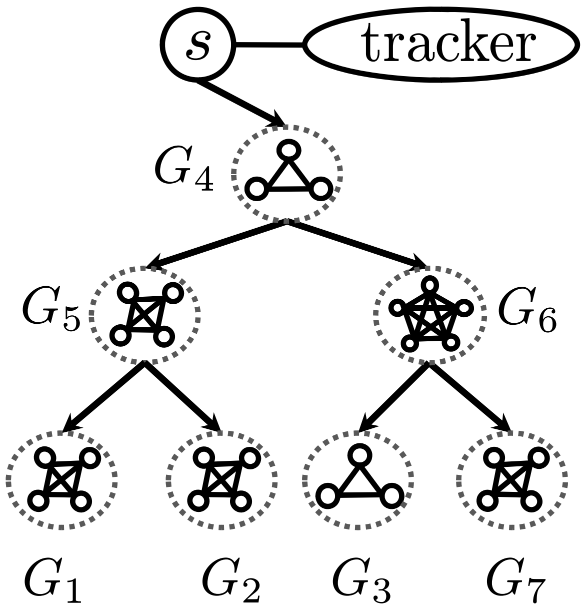
- Theorem [Liu-Chen-Sengupta-Chiang-Li-Chou 10]: let r_{Bubble} be the streaming rate achieved by Bubble algorithm, and $\bar{r}(M)$ be the streaming capacity under node degree bound M. We have

$$r_{Bubble} \geq \frac{1}{2}\bar{r}(M)$$

How to Do Better? Create Homogeneity!

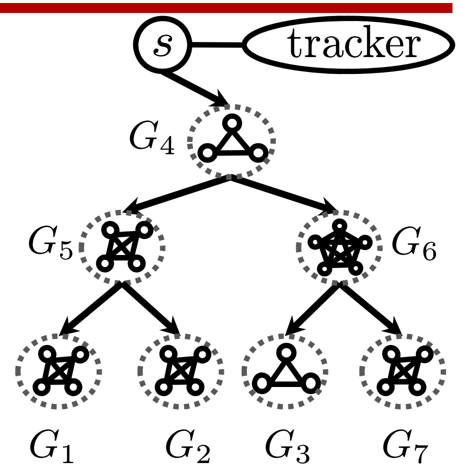
- Group O(log N) peers to create homogeneous clusters
  - "upload capacity" of a cluster: average peer capacity inside the cluster
  - By CLT, clusters' upload capacity are roughly the same



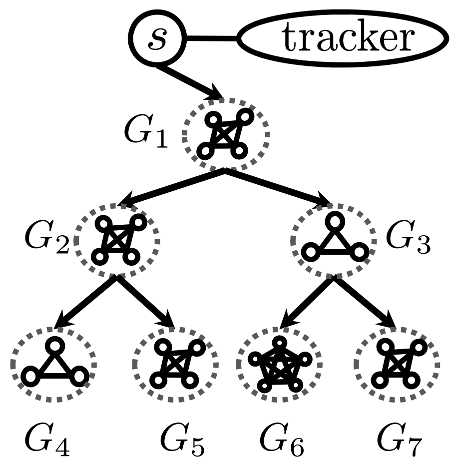


Cluster-Tree Algorithm [Liu-Chen-Sengupta-Chiang-Li-Chou 10]

- Inside each cluster
  - Use dense MutualCast trees to deliver content locally
  - Take care of peer heterogeneity locally



- Across clusters
  - Use sparse CoopNet/SplitStream trees to deliver content globally
  - Efficient content delivery across trees



Cluster-Tree: Performance Guarantee

- Theorem [Liu-Chen-Sengupta-Chiang-Li-Chou 10]: If node degree bound M = O(log N), then

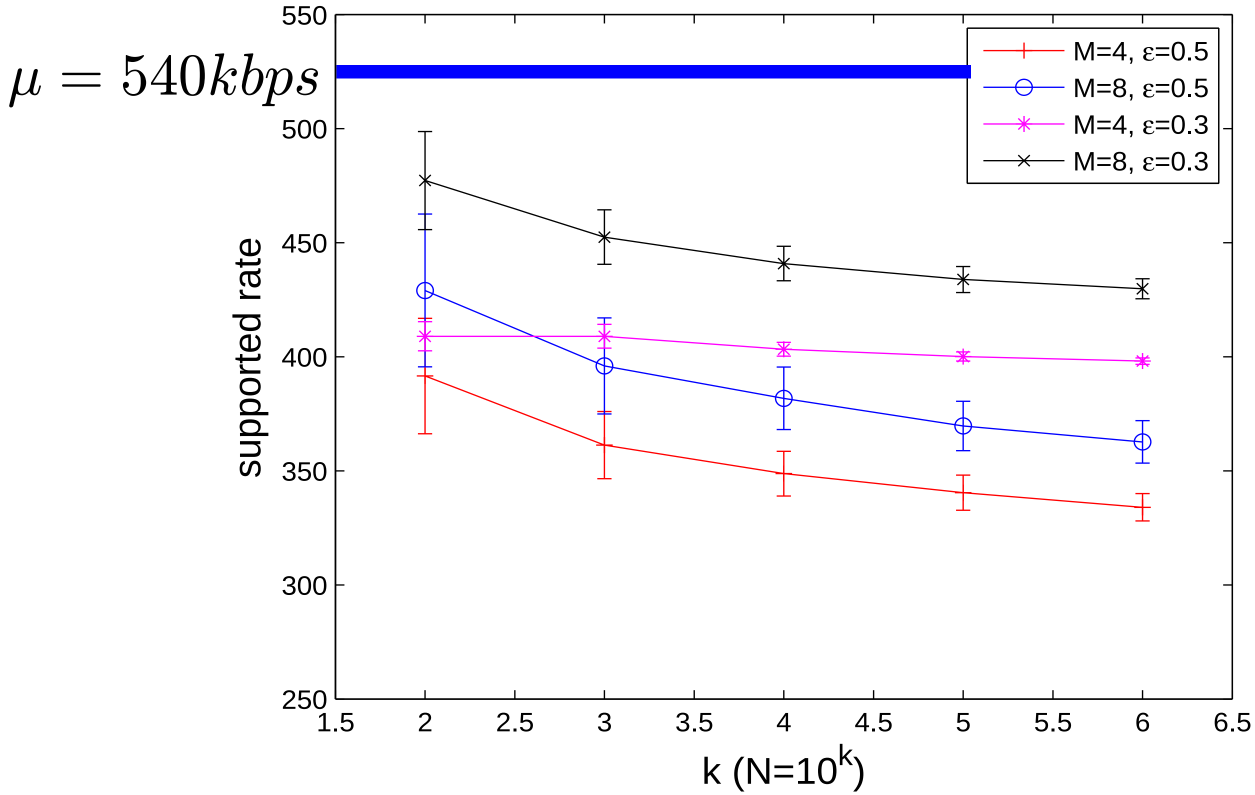
$$r_{Cluster-Tree} \geq (1-\epsilon)\text{Capacity}$$

with high probability, where $\epsilon > 0$ is constant.

- **Insight:**
  - Randomly peering in a locally dense and globally sparse manner is good
  - O(log N) neighbors per peer is enough

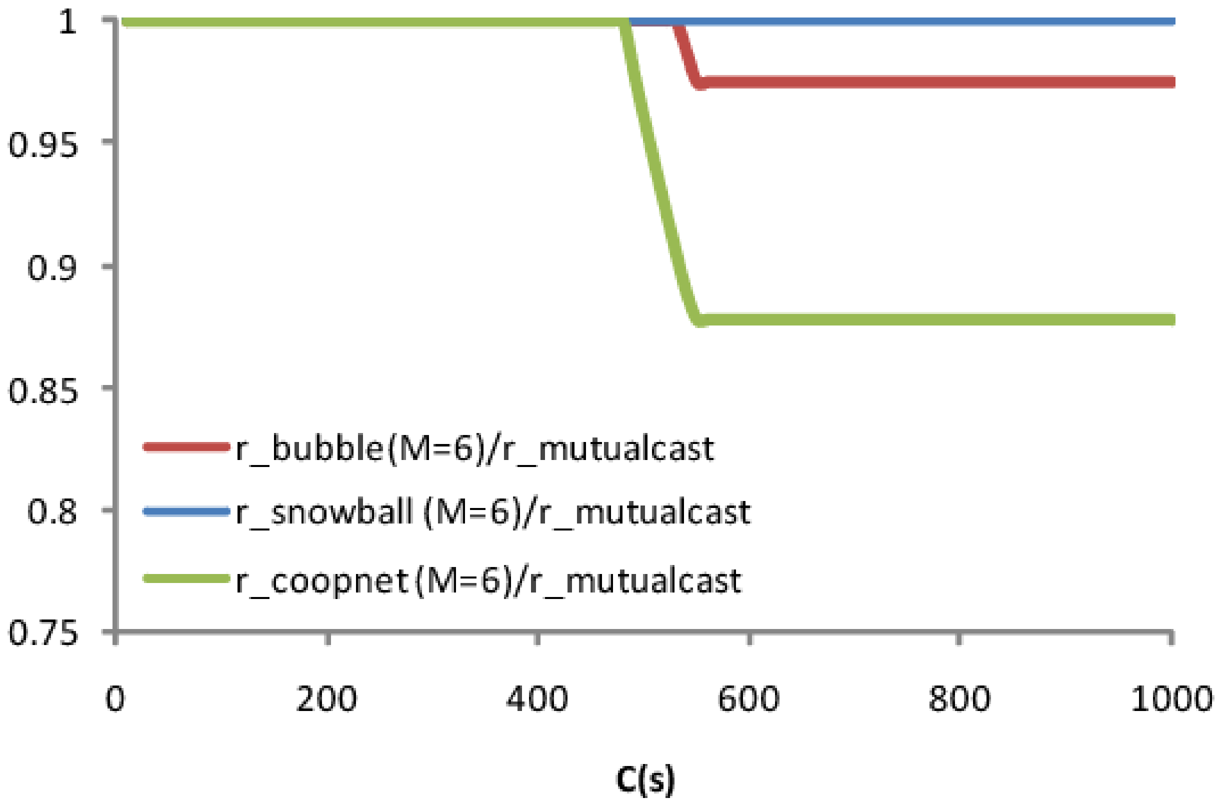
Simulation: Cluster-Tree Algorithm

- Peer upload capacities from trace statistics
- Peer node degree: 86 when N = 1 Million nodes



Simulation: Bubble Algorithm

- Peer upload capacities from trace statistics
- Bubble achieves high streaming rate



Big Picture

	general graph	arbitrary node degree bound	optimality (exact or 1-\epsilon)	distributed
Li-Chou-Zhang 05 (Mutualcast), Kumar-Ross 07, Massoulie et al. 07	×	×	√	√
Coopnet/SplitStream	×	√	×	×
ZIGZAG, PRIME, PPLIVE, UUSEE and most commercial systems	√	√	×	√
Iterative by Sengupta-Liu-Chen-Chiang-Li-Chou 09	√	×	√	×
Cluster-tree	×	√	Optimal if degree bound is O(ln N)	×
Work coming up	√	√	√	√

II. QoS in Static Peer-to-Peer Systems
B. Streaming Delay

Chunk Based P2P Streaming Delay Minimization

- Mesh is multiple short-time lived trees (from a single chunk's viewpoint)
- A video stream consists of infinitely many chunks, that exploit exponential # of trees
- Question
  - How to construct a multi-tree that minimizes worst user delay for the stream, under node degree bound?
  - Can we achieve maximum streaming rate and minimum worst delay simultaneously?

Big Picture

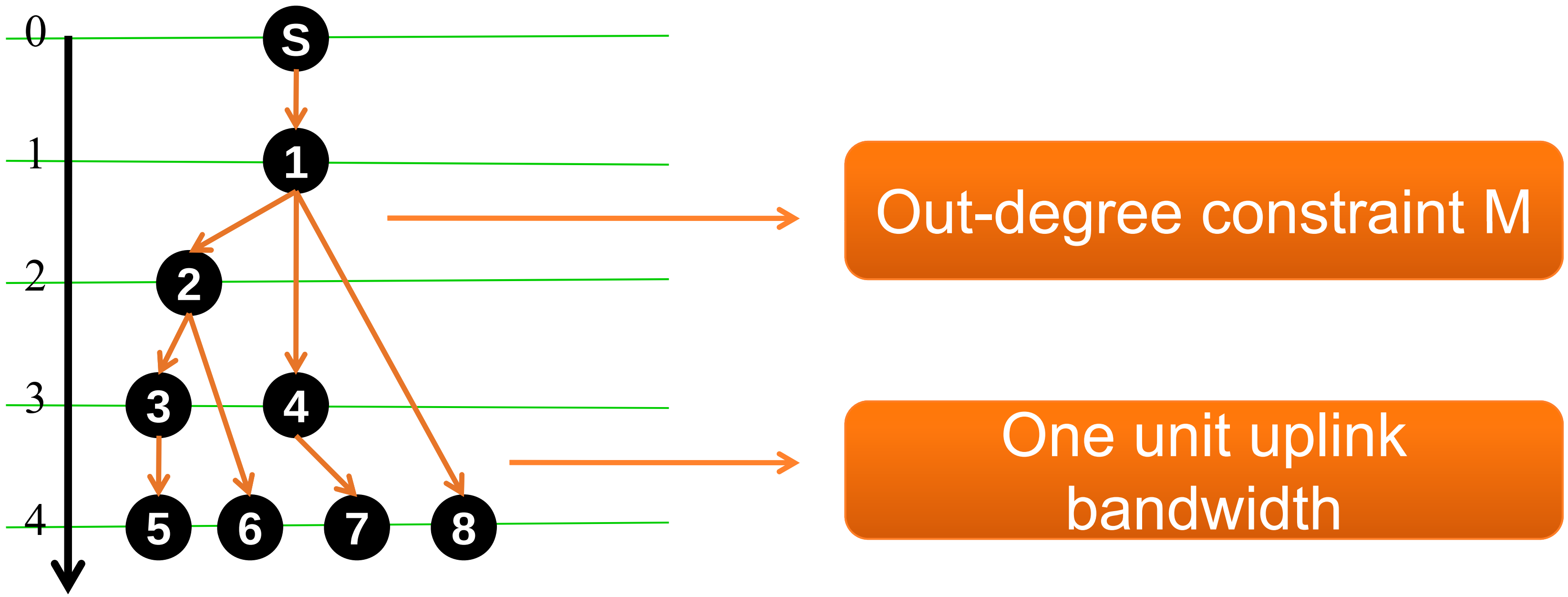
	Homogeneous	Heterogeneous
Singe Chunk No degree-bound	[Yong 07] O(logN)	[Jiang-Zhang-Chen-Chiang 10]
Single Chunk Degree-bounded	[Bianchi-Melazzi-Bracciale-Piccolo-Salsano 09]	Open
Streaming No degree-bound	[Jiang-Zhang-Chen-Chiang 10]	Partially solved
Streaming Degree-bounded	[Jiang-Zhang-Chen-Chiang 10]	Open

Achieving Streaming Capacity and Delay Bound Simultaneously

- *In a homogeneous P2P system where peers have unit upload capacities, for arbitrary population N, arbitrary out-degree bound M, we achieve simultaneously*
  - *optimal streaming rate 1*
  - *optimal max-user delay log(N+1/M)+c*
  - *by packing a finite number of (logN) trees*

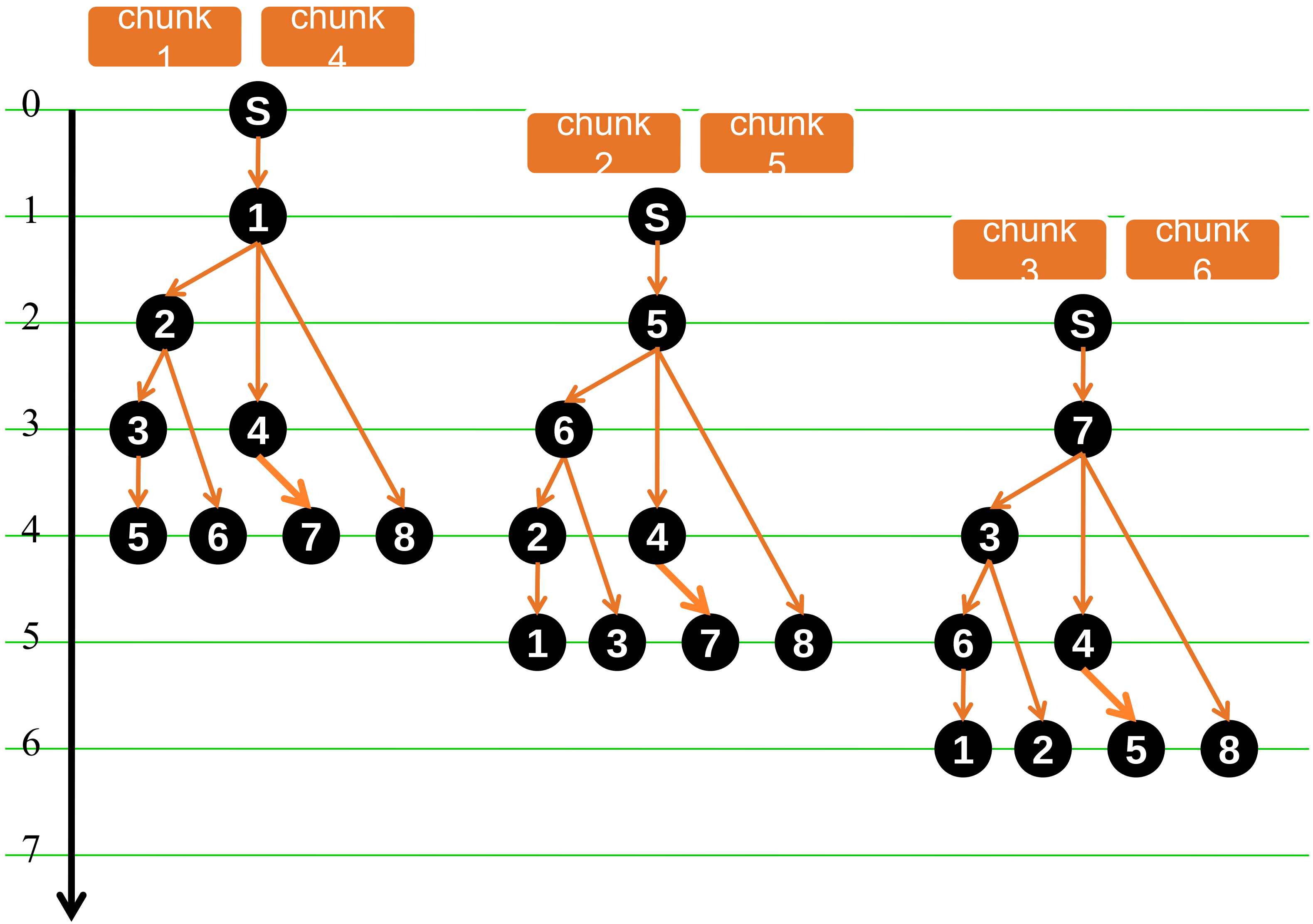
Minimum Delay: The Single-chunk Case

- Motivated by an M-step Fibonacci sequence [Bianchi-Melazzi-Bracciale-Piccolo-Salsano 09]
- A building block for multiple chunks (continuous stream)

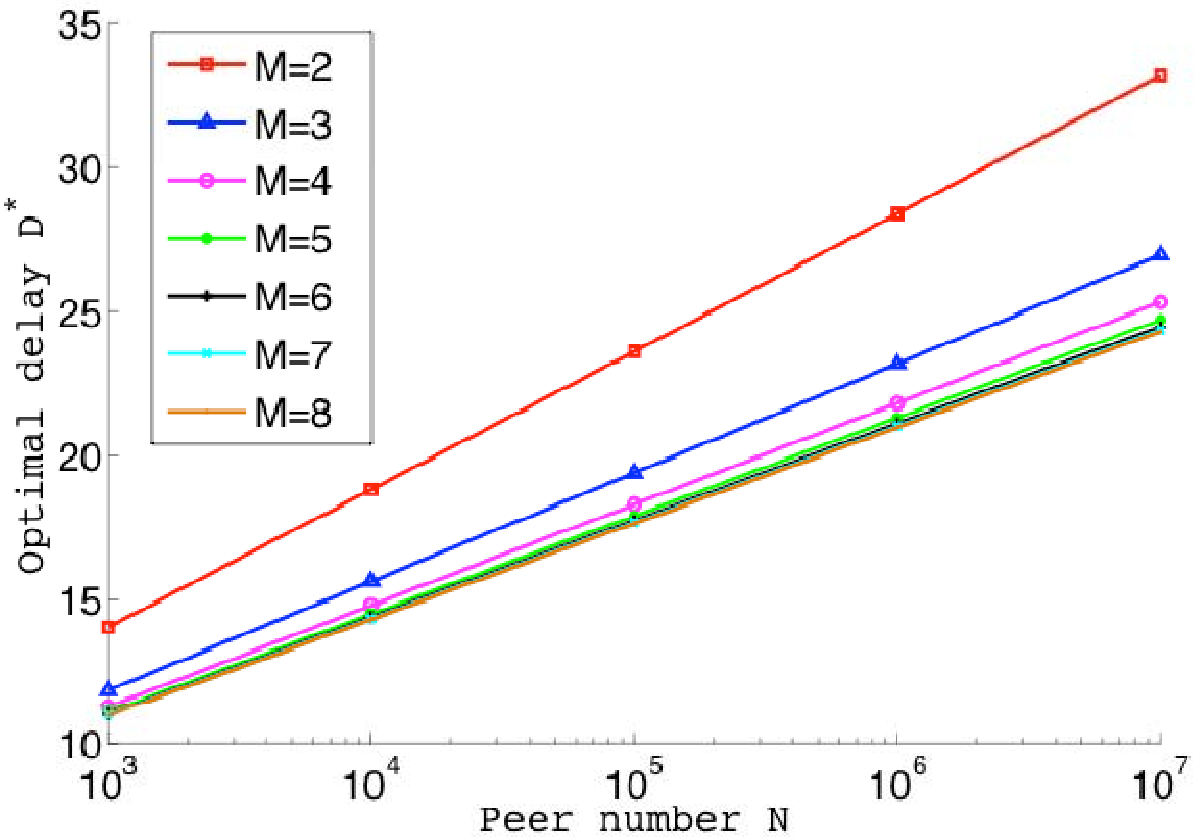


N=8, M=3

Minimum Delay: The Multi-chunk Case [Jiang-Zhang-Chen-Chiang 10]



A Small Out-degree Is Enough for Small Delay



An out-degree of 8 achieves minimum delay in practical system design

III. Peer-to-Peer Video-on-Demand (VoD) Systems

Outline

Y. Huang, et al., "Challenges, Design and Analysis of a Large-scale P2P-VoD System", ACM SIGCOMM 2008.

[Acknowledgement: Slides taken from authors' Sigcomm presentation]

- ❖Architecture of a PPLive P2P-VoD system
- ❖Performance metrics
- ❖Measurement results and analysis
- ❖Conclusions

P2P Overview

- Advantages of P2P
  - Users help each other so that the server load is significantly reduced.
  - P2P increases robustness in case of failures by replicating data over multiple peers.
- P2P services
  - P2P file downloading : BitTorrent and Emule
  - P2P live streaming : Coolstreaming, PPStream and PPLive
  - P2P video-on-demand (P2P-VoD) : Joost, GridCast, PFSVOD, UUSee, PPStream, PPLive...

P2P-VoD System Properties

- Less synchronous compared to live streaming
  - Like P2P streaming systems, P2P-VoD systems also deliver the content by streaming, but peers can watch different parts of a video at the same time.
- Requires more storage
  - P2P-VoD systems require each user to contribute a small amount of storage (usually 1GB) instead of only the playback buffer in memory as in the P2P streaming system.
- Requires careful design of mechanisms for
  - Content Replication
  - Content Discovery
  - Peer Scheduling

P2P-VoD system

- Servers
  - The source of content
- Trackers
  - Help peers connect to other peers to share the content
- Bootstrap server
  - Helps peers to find a suitable tracker
- Peers
  - Run P2P-VoD software
  - Implement DHT(Dynamic Hash Table)
- Other servers
  - Log servers : log significant events for data measurement
  - Transit servers : help peers behind NAT boxes

Design Issues To Be Considered

- Segment size
- Replication strategy
- Content discovery
- Piece selection
- Transmission Strategy
- Others:
  - NAT and Firewalls
  - Content Authentication

Segment Size

- What is a suitable segment size?
  - Small
    - More flexibility of scheduling
    - But larger overhead
      - Header overhead
      - Bitmap overhead
      - Protocol overhead
  - Large
    - Smaller overhead
    - Limited by viewing rate
- Segmentation of a movie in PPLive's VoD system

Segment	Designed for	Size
movie	entire video	$>$ 100MB
chunk	unit for storage and advertisement	2MB
piece	unit for playback	16KB
sub-piece	unit for transmission	1KB

Table 1: Different units of a movie

Replication Strategy

- Goal
  - To make the chunks as available to the user population as possible to meet users' viewing demand
- Considerations
  - Whether to allow multiple movies be cached
    - Multiple movie cache (MVC) - more flexible for satisfying user demands
      - **PPLive uses MVC**
    - Single movie cache (SVC) - simple
  - Whether to pre-fetch or not
    - Improves performance
    - Unnecessarily wastes uplink bandwidth
    - In ADSL, upload capacity is affected if there is simultaneous download
    - Dynamic peer behavior increases risk of wastage
      - **PPLive chooses not to pre-fetch**

Replication Strategy(Cont.)

- Remove chunks or movies?
  - **PPLive marks entire movie for removal**
- Which chunk/movie to remove
  - Least recently used (LRU) –**Original choice of PPLive**
  - Least frequently used (LFU)
  - Weighted LRU:
    - How complete the movie is already cached locally?
    - How needed a copy of movie is ATD (Available To Demand)
      - ATD = c/n

        where, c = number of peers having the movie in the cache, n = number of peers watching the movie
      - The ATD information for weight computation is provided by the tracker.
      - In current systems, the average interval between caching decisions is about 5 to 15 minutes.
      - It improves the server loading from 19% down to a range of 11% to 7%.

Content Discovery

- Goal : discover the content they need and which peers are holding that content with the minimum overhead.
- Trackers
  - Used to keep track of which peers have the movie
  - User informs tracker when it starts watching or deletes a movie
- Gossip method
  - Used to discover which chunks are with whom
  - Makes the system more robust
- DHT
  - Used to automatically assign movies to trackers
  - Implemented by peers to provide a non-deterministic path to trackers
    - Originally DHT is implemented by tracker nodes

Piece Selection

- Which piece to download first
  - Sequential
    - Select the piece that is closest to what is needed for the video playback
  - Rarest first
    - Select the rarest piece help speeding up the spread of pieces, hence indirectly helps streaming quality.
  - Anchor-based
    - When a user tries to jump to a particular location in the movie, if the piece for that location is missing then the closest anchor point is used instead.

PPLive gives priority to sequential first and then rarest-first

Transmission Strategy

- Goals
  - Maximize (to achieve the needed) downloading rate
  - Minimize the overheads, dud to duplicated transmissions and requests
- Strategies
  - A peer can work with one neighbor at a time.
  - Request the same content from multiple neighbors simultaneously
  - Request the different content from multiple neighbors simultaneously, when a request times out, it is redirected to a different neighbor; **PPLive uses this scheme**
    - For playback rate of 500Kbps, 8~20 neighbors is the best; playback rate of 1Mbps, 16~32 neighbors is the best.
    - When the neighboring peers cannot supply sufficient downloading rate, the content server can always be used to supplement the need.

Other Design Issues

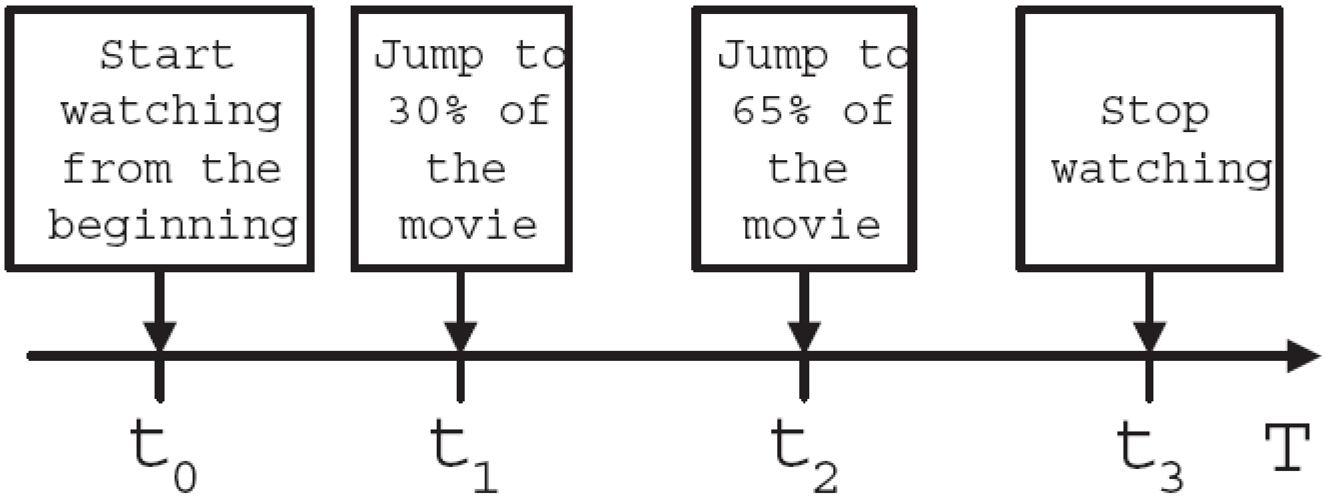
- NAT
  - Discovering different types of NAT boxes
    - *Full Cone* NAT, *Symmetric* NAT, *Port- restricted* NAT…
  - About 60%-80% of peers are found to be behind NAT
- Firewall
  - PPLive software carefully pace the upload rate and request rate to make sure the firewalls will not consider PPLive peers as malicious attackers
- Content authentication
  - Authentication by message digest or digital signature

Measurement Metrics

- User behavior
  - User arrival patterns
  - How long they stayed watching a movie
  - Used to improve the design of the replication strategy
- External performance metrics
  - User satisfaction
  - Server load
  - Used to measure the system performance perceived externally
- Health of replication
  - Measures how well a P2P-VoD system is replicating a content
  - Used to infer how well an important component of the system is doing

User Behavior-MVR (Movie Viewing Record)

User ID	Movie ID	Start time	End time	Start pos.



	UID	MID	ST	ET	SP
MVR1:	U1	M1	t_0	t_1	0%
MVR2:	U1	M1	t_1	t_2	30%
MVR3:	U1	M1	t_2	t_3	65%

Figure 1: Example to show how MVRs are generated

User Satisfaction

- Simple fluency
  - Fraction of time a user spends watching a movie out of the total viewing time (waiting and watching time for that movie)
  - Fluency *F(m,i)* for a movie *m* and user *i*

$$F(m,i) = \frac{\sum_{r \in R(m,i)} (r(ET) - r(ST) - r(BT))}{\sum_{r \in R(m,i)} (r(ET) - r(ST))}. \qquad (1)$$

R(*m*, *i*) : the set of all MVRs for a given movie *m* and user *i*

n(*m*, *i*) : the number of MVRs in *R*(*m*, *i*)

r : one of the MVRs in *R*(*m*, *i*)

BT : Buffering Time, ST : Starting Time, ET : Ending Time, and SP : Starting Position

User Satisfaction (Cont1.)

- User satisfaction index
  - Considers the quality of the delivery of the content

$$S(m,i) = \sum_{k=1}^{n(m,i)} W_k r_k(Q). \qquad (3)$$

r(Q) : a grade for the average viewing quality for an MVR *r*

$$W_k = \frac{(r_k(ET) - r_k(ST) - r_k(BT))}{\sum_{r \in R(m,i)} (r(ET) - r(ST))}$$

User Satisfaction (Cont2.)

- In Fig. 1, assume there is a buffering time of 10 (time units) for each MVR. The **fluency** can be computed as:

$$F = \frac{(t_1 - t_0 - 10) + (t_2 - t_1 - 10) + (t_3 - t_2 - 10)}{(t_3 - t_0)}$$

- Suppose the user grade for the three MVR were 0.9, 0.5, 0.9 respectively. Then the **user satisfaction index** can be calculated as:

$$S = \frac{0.9(t_1 - t_0 - 10) + 0.5(t_2 - t_1 - 10) + 0.9(t_3 - t_2 - 10)}{(t_3 - t_0)}$$

Health of Replication

- Health index : use to reflect the effectiveness of the content replication strategy of a P2P-VoD system.
- The health index (for replication) can be defined at 3 levels:
  - Movie level
    - The number of active peers who have advertised storing chunks of that movie
    - Information about that movie collected by the tracker
  - Weighted movie level
    - Considers the fraction of chunks a peer has in computing the index
    - If a peers stores 50 percent of a movie, it is counted as 0.5
  - Chunk bitmap level
    - The number of copies of each chunk of a movie is stored by peer
    - Used to compute other statistics
      - The average number of copies of a chunk in a movie, the minimum number of chunks, the variance of the number of chunks.

Measurement

- All these data traces were collected from 12/ 23/2007 to 12/29/2007
- Log server : collect various sorts of measurement data from peers.
- Tracker : aggregate the collected information and pass it on to the log server
- Peer : collect data and do some amount of aggregation, filtering and pre-computation before passing them to the log server
- We have collected the data trace on 10 movies from the P2P-VoD log server
- Whenever a peer selects a movie for viewing, the client software creates the MVRs and computes the viewing satisfaction index, and these information are sent to the log server
- Assume the playback rate is about 380kbps
- To determine the most popular movie, we count only those MVRs whose starting position (SP) is equal to zero (e.g., MVRs which view the movie at the beginning)
  - Movie 2 is the most popular movie with 95005 users
  - Movie 3 is the least popular movie with 8423 users

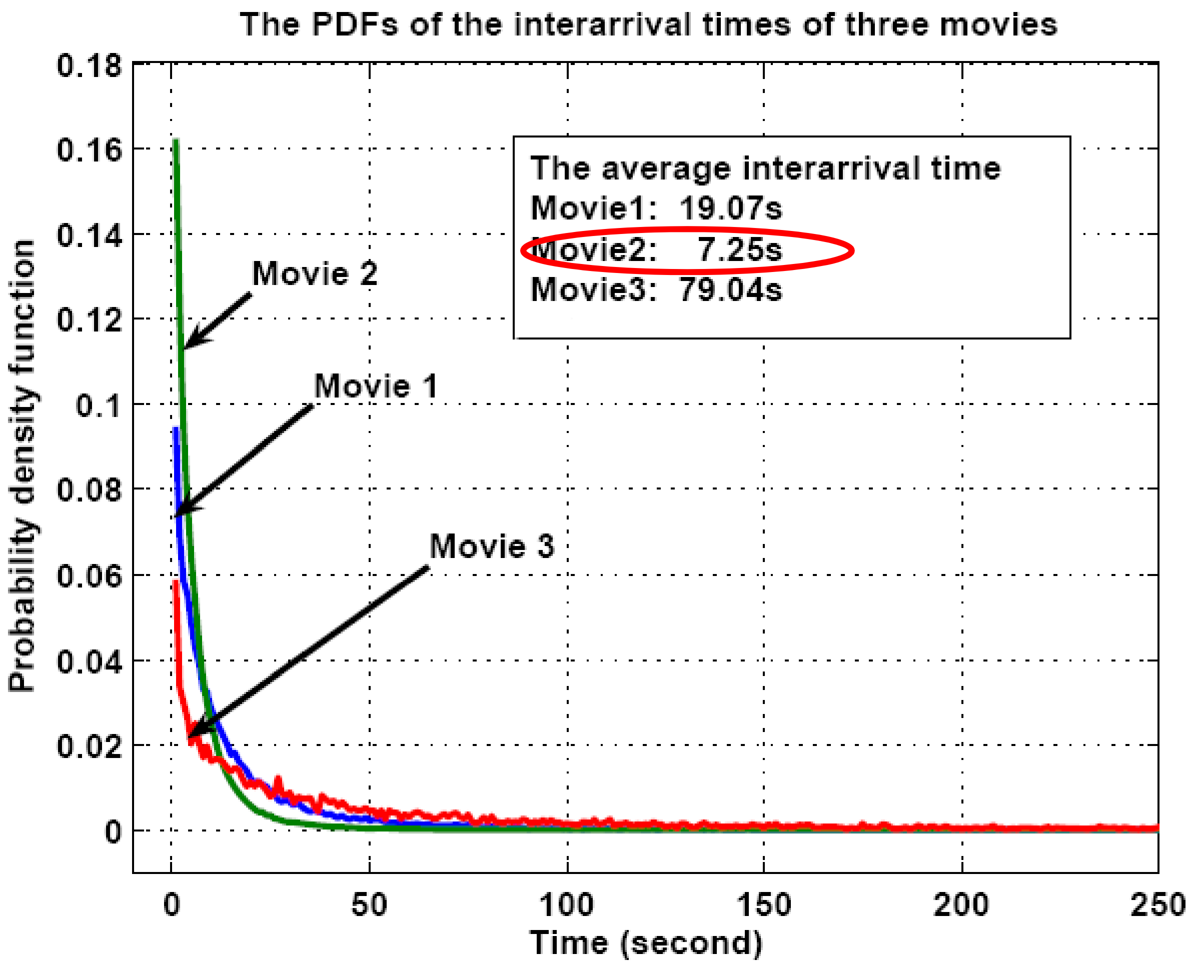
Statistics on video objects

- Overall statistics of the 3 typical movies

Movie Index:	Movie 1	Movie 2	Movie 3
Total Length (in sec)	5100s	2820s	6600s
No. of Chunks	121	67	151
Total No. of MVRs	56157	322311	15094
Total No. of MVRs with Start Position = 0 (or # of unique viewers)	35160	95005	8423
Ave. # of Jump	1.6	3.4	1.8
Ave. viewing Duration for a MVR	829.8s	147.6s	620.2s
Normalized viewing Duration (normalized by the movie duration)	16.3%	5.2%	9.4%

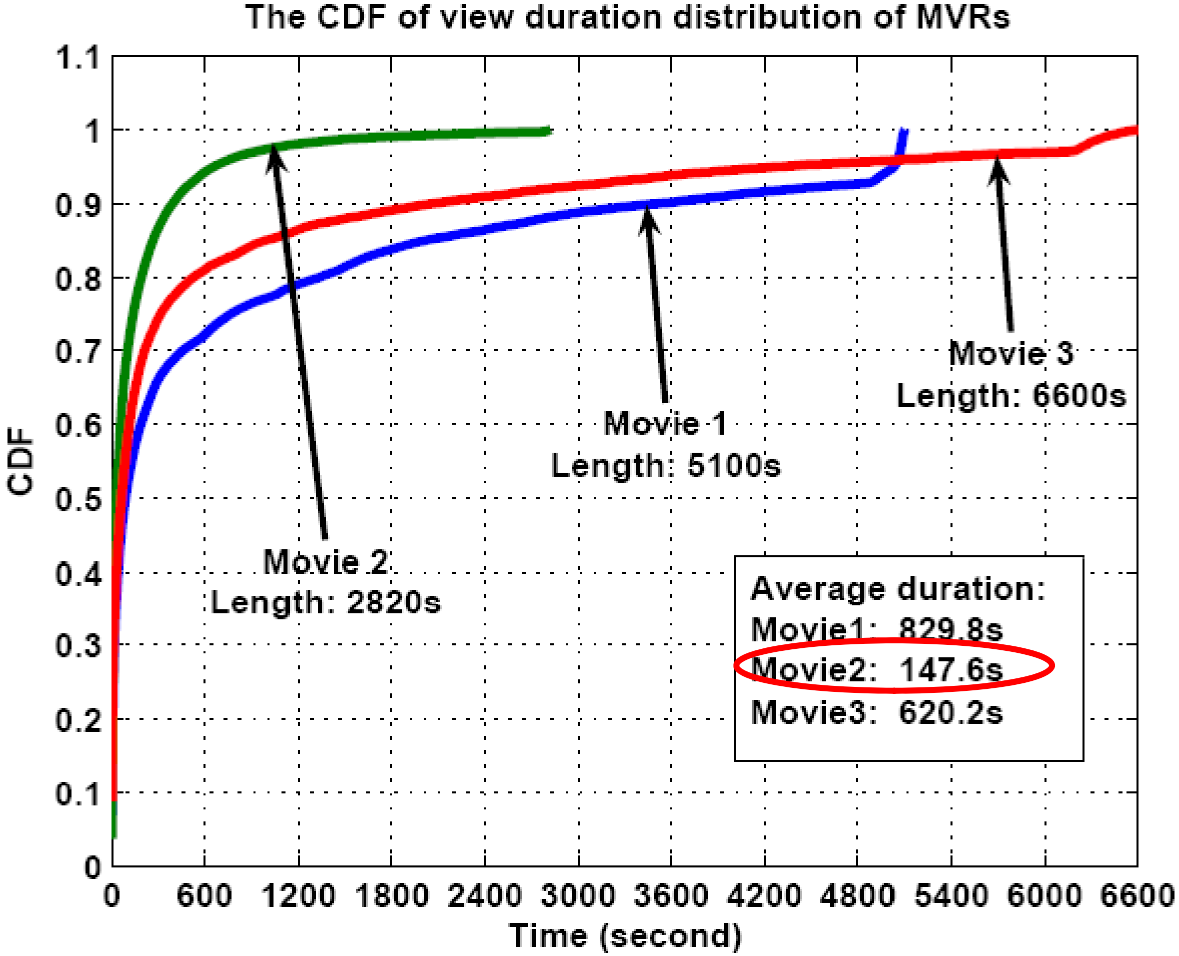
Table 3: Overall statistics of the three typical movies.

Statistics on user behavior (1) : Interarrival time distribution of viewers



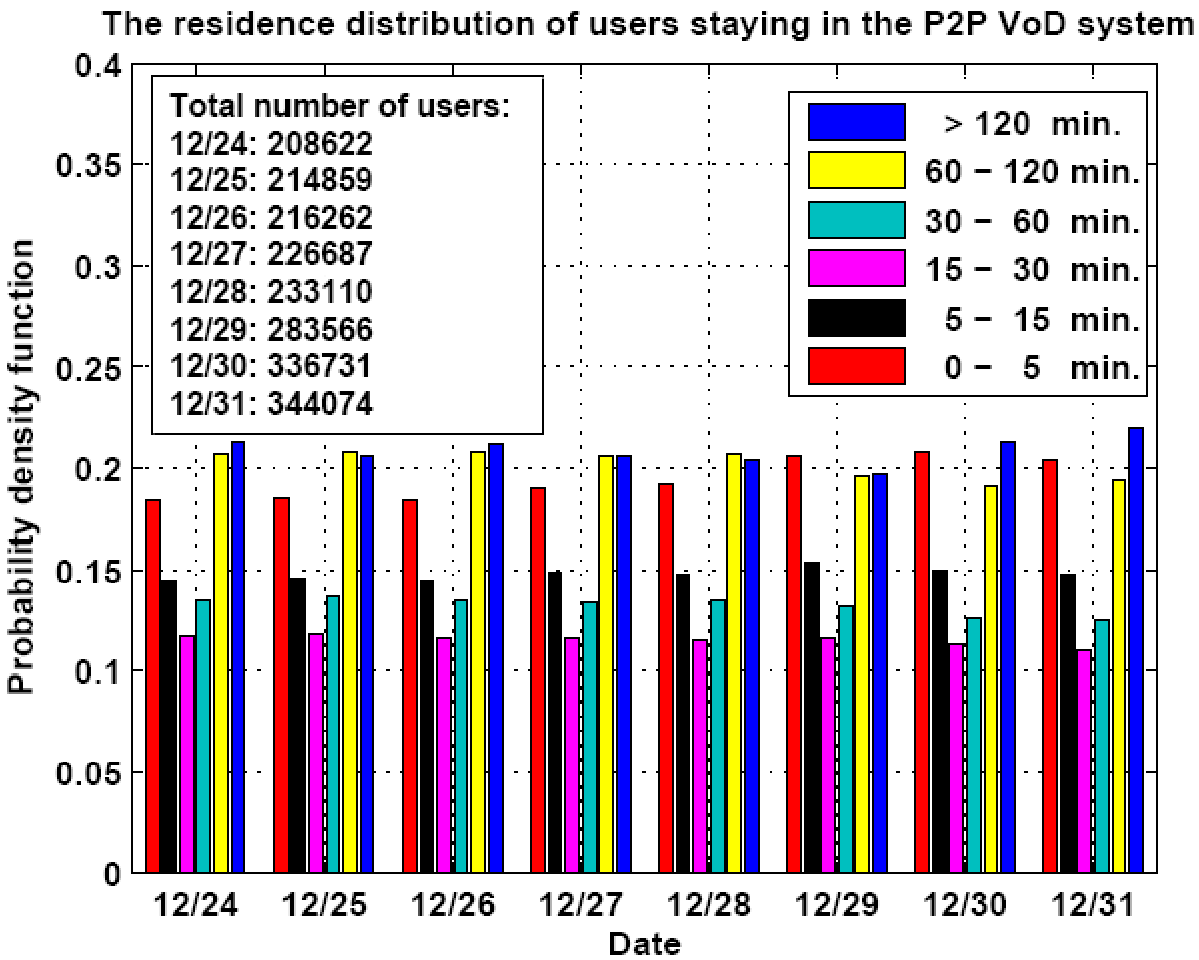
Interarrival times of viewers : the differences of the ST fields between to consecutive MVRs

Statistics on user behavior (2) : View duration distribution



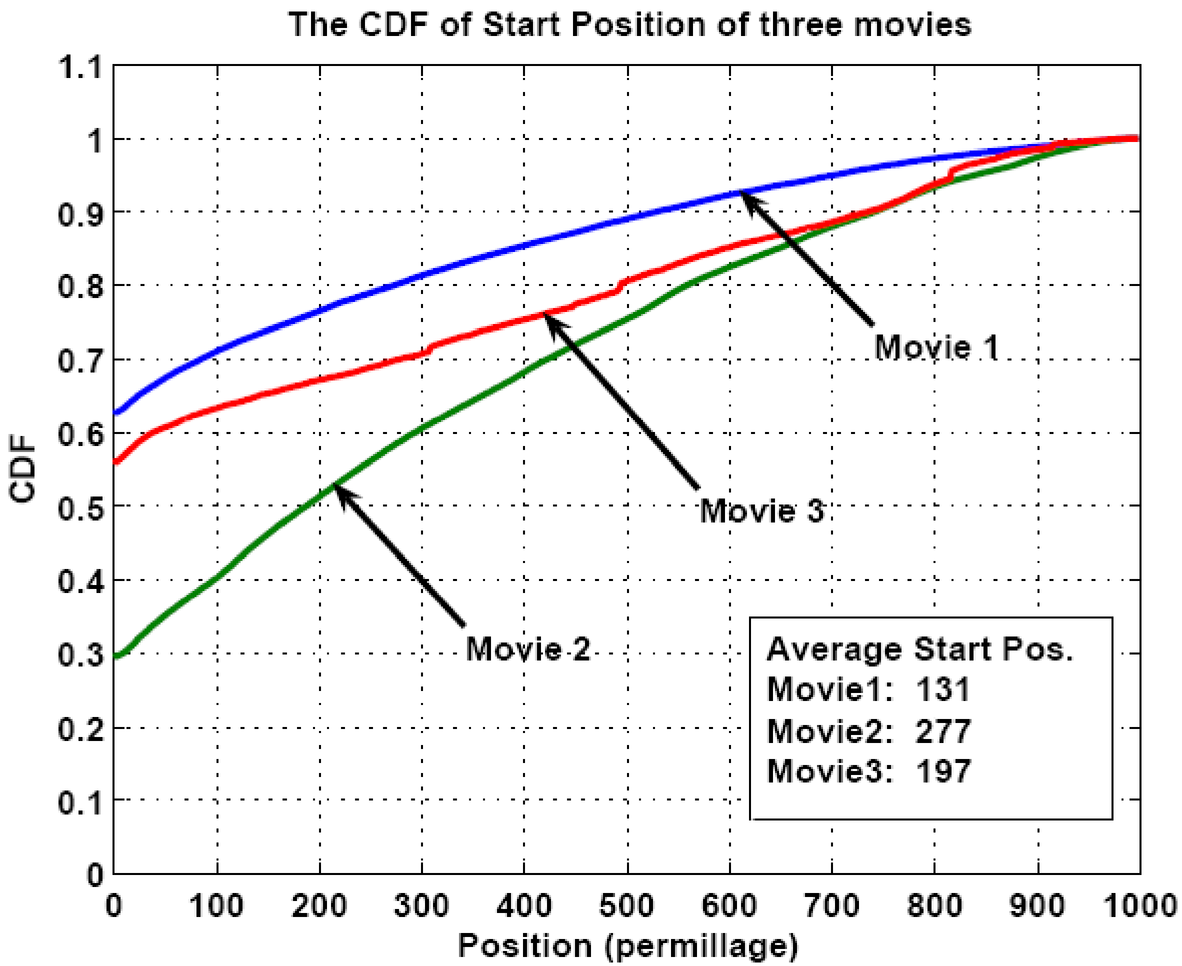
Very high percentage of MVRs are of short duration (less than 10 minutes). This implies that for these 3 movies, the viewing stretch is of short duration with high probability.

Statistics on user behavior (3) : Residence distribution of users



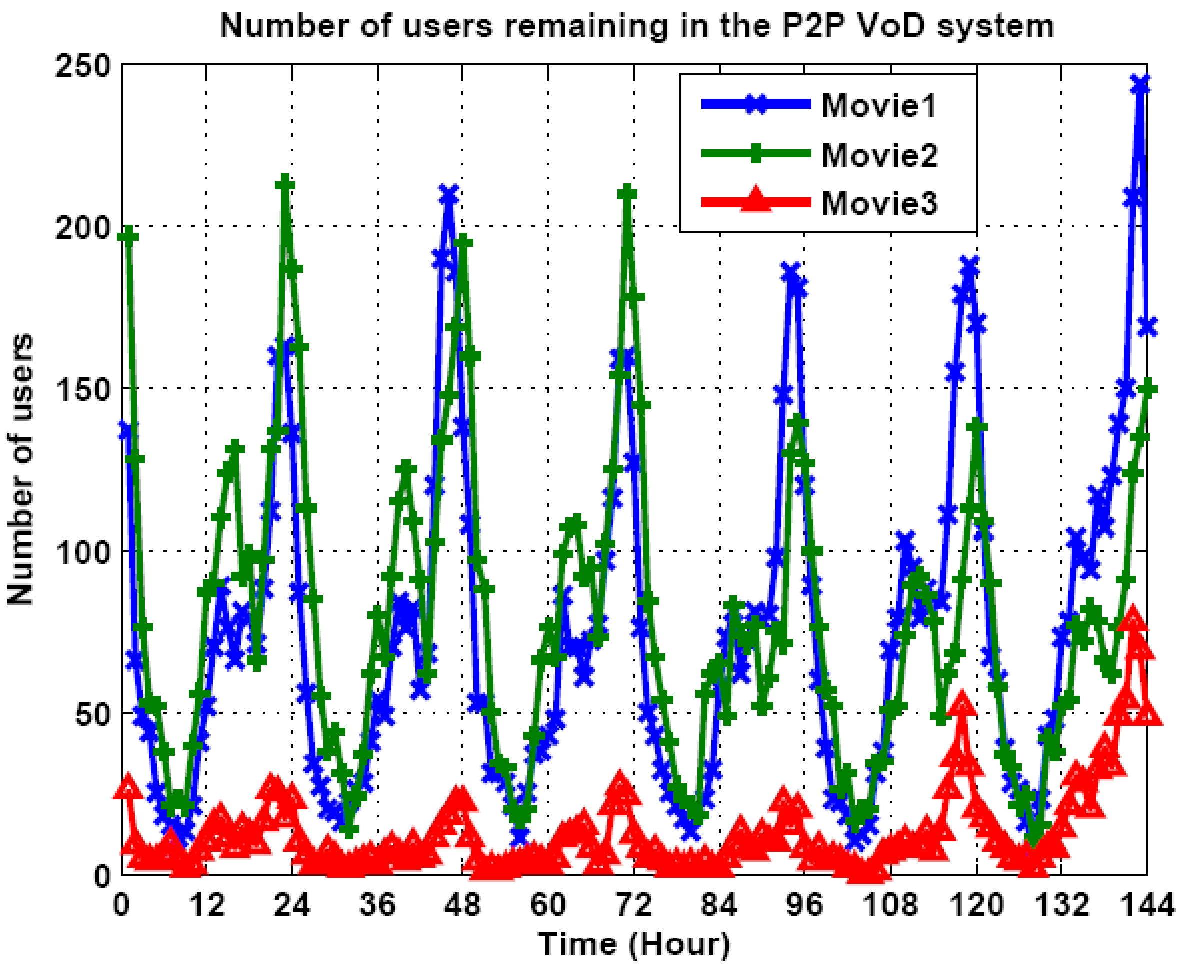
There is a high fraction of peers (over 70%) which stays in the P2P-VoD system for over 15 minutes, and these peers provide upload services to the community.

Statistics on user behavior (4): Start position distribution



Users who watch Movie 2 are more likely to jump to some other positions than users who watch Movie 1 and 3

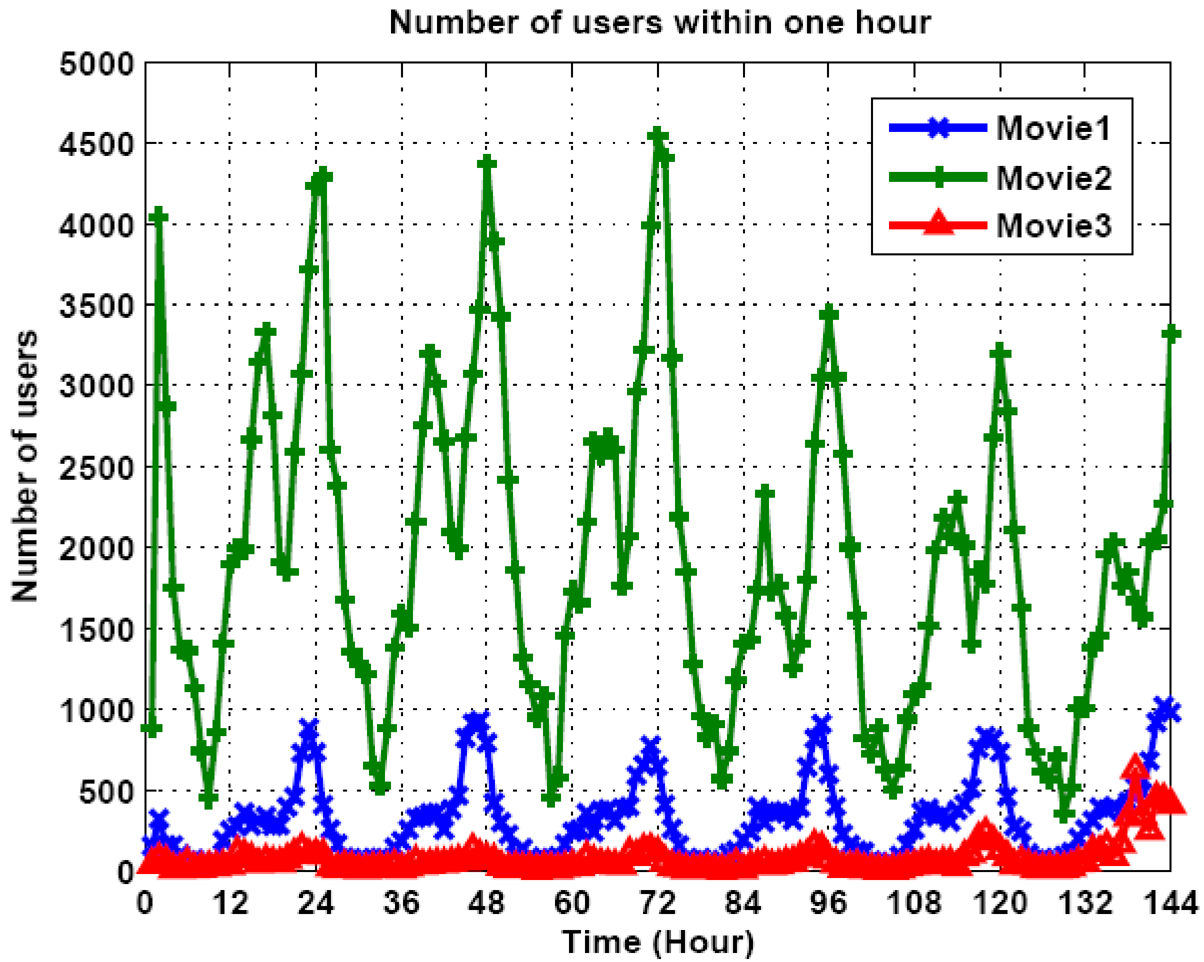
Statistics on user behavior (5): Number of viewing actions



- The total number of viewing activities (or MVRs) at *each* sampling time point.
- *"daily periodicity"* of user behavior. There are two daily peaks, which occur at around 2:00 P.M. and 11:00 P.M.

Figure 7: Number of viewing actions at each hourly sampling point (6 days measurement).

Statistics on user behavior (5): Number of viewing actions(Cont.)

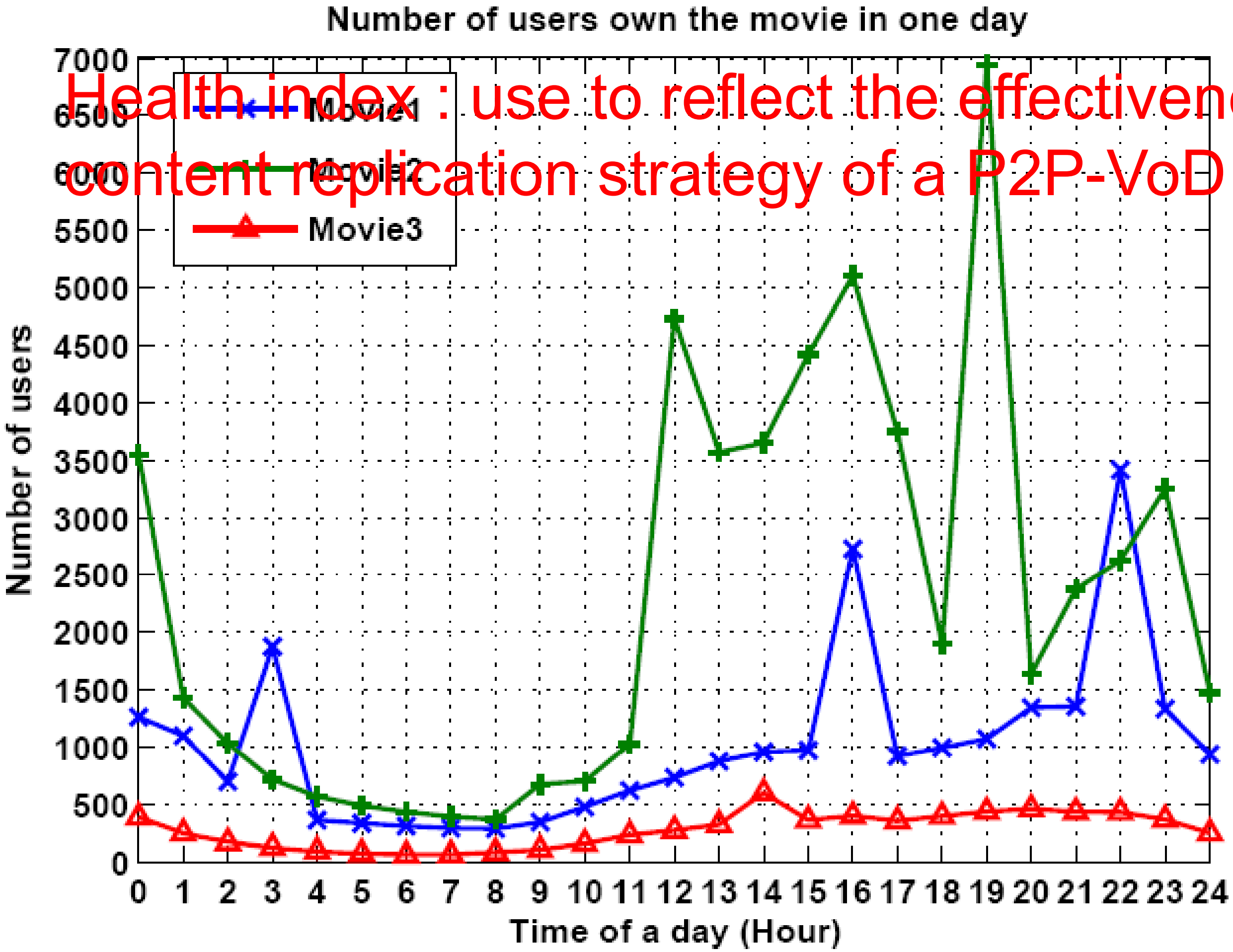


- The total number of viewing activities (or MVRs) that occurs *between* two sampling points.
- *"daily periodicity"* of user behavior. There are two daily peaks, which occur at around 2:00 P.M. and 11:00 P.M.

Figure 8: Total number of viewing actions within each sampling hour(6 days measurement).

Health index of Movies (1): Number of peers that own the movie

Health index : use to reflect the effectiveness of the content replication strategy of a P2P-VoD system.



- Owning a movie implies that the peer is still in the P2P-VoD system.
- Movie 2 being the most popular movie.
- The number of users owning the movie is lowest during the time frame of 5:00 A.M. to 9:00 A.M.

Figure 9: Number of users owning at least one chunk of the movie at different time points.

Health index of Movies (2)

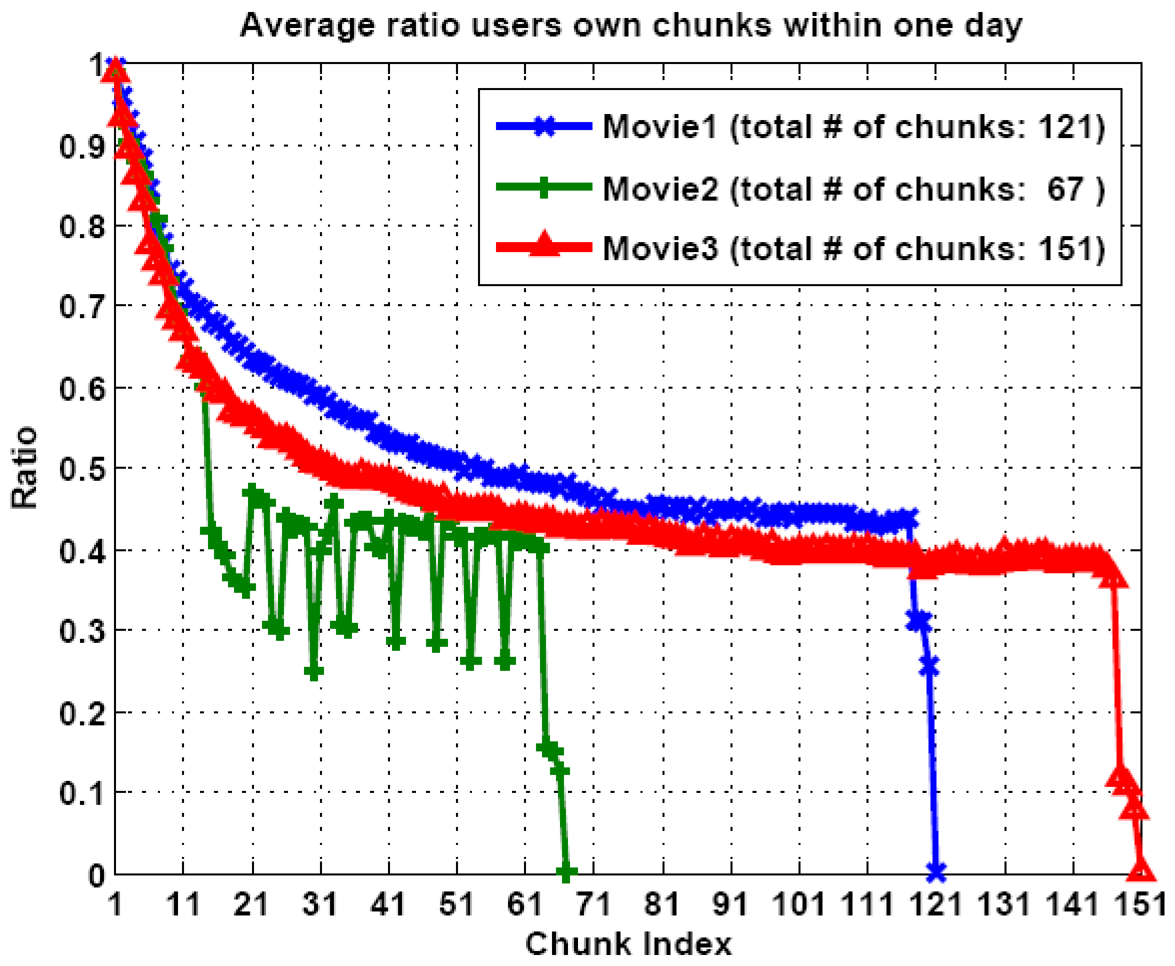


Figure 10: Average owning ratio for all chunks in the three movies.

unks

- The health index for "early" chunks is very good.

at time t

time t

- Many peers may browse through the beginning of a movie.

y of chunk i in

- The health index is still acceptable since at least 30% of the peers have those chunks.

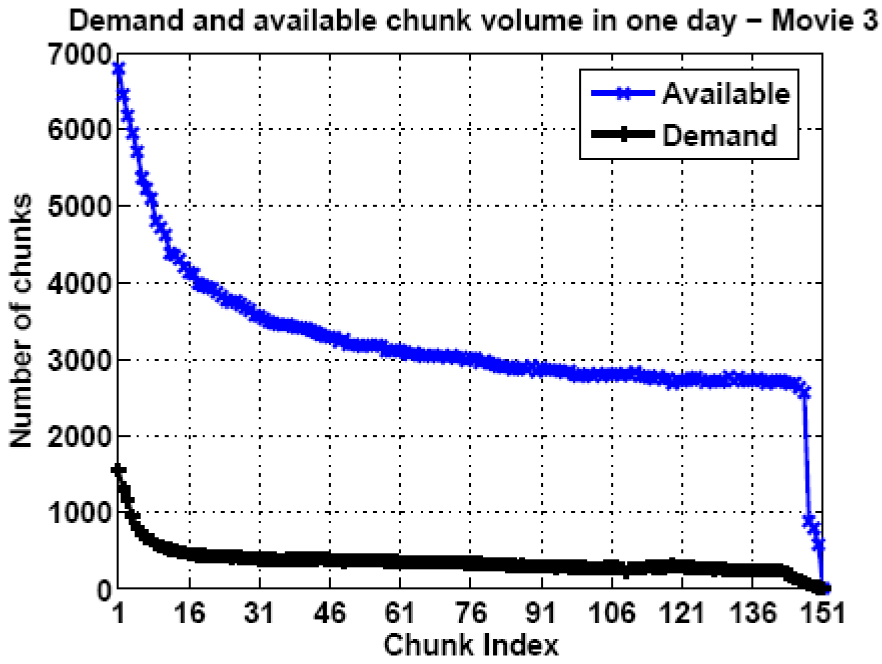
Health index of Movies (3)

(a) The health index for these 3 movies are very good since the number of replicated chunk is much higher than the workload demand.

(b) The large fluctuation of the chunk availability for Movie 2 is due to the high interactivity of users.

(c) Users tend to skip the last chunk of the movie.

❖ Chunk availability and chunk demand



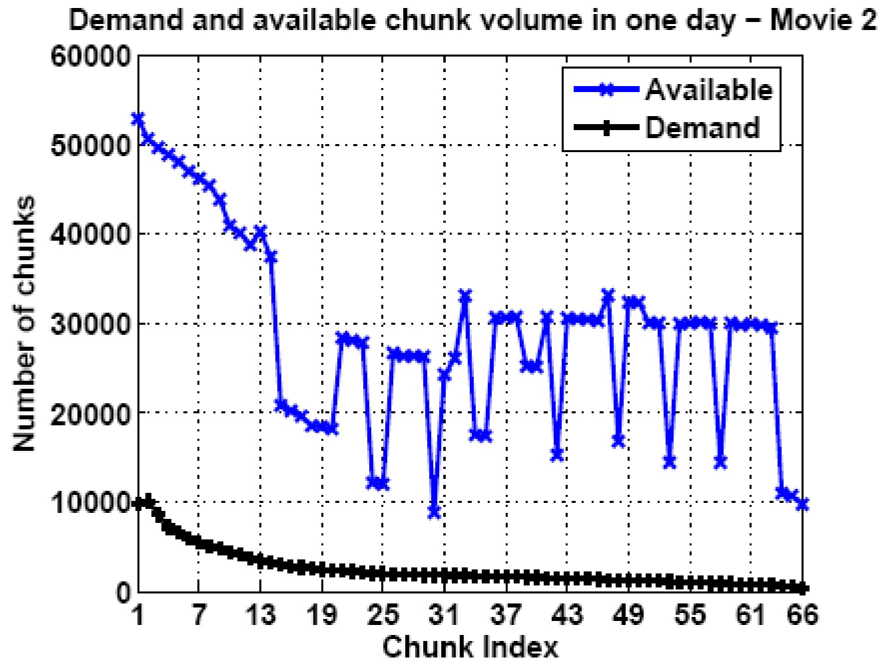
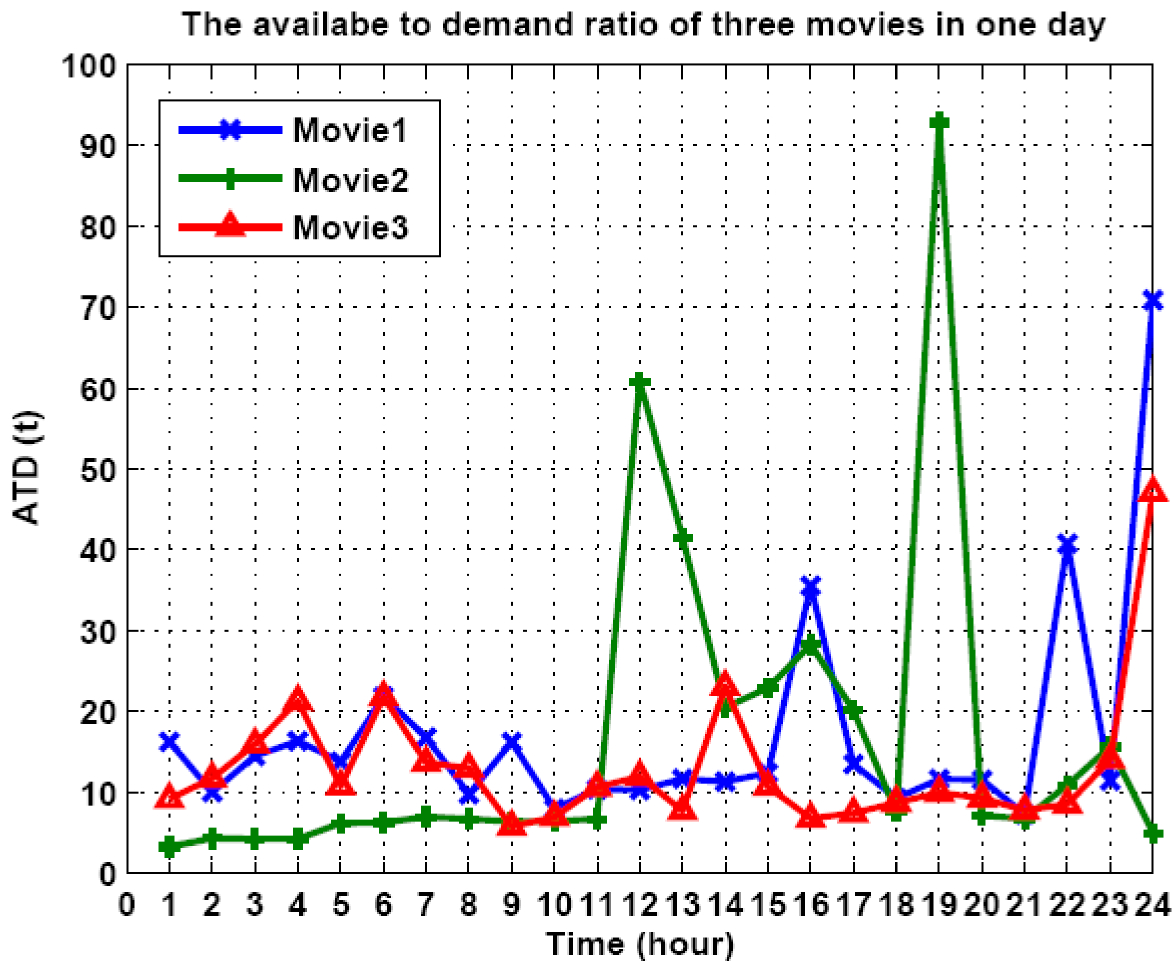


Figure 11: Comparison of number replicated chunks and chunk demand of 3 movies in one day (from 0:00 to 24:00 January 6, 2008).

Health index of Movies (4): ATD (Available To Demand) ratios



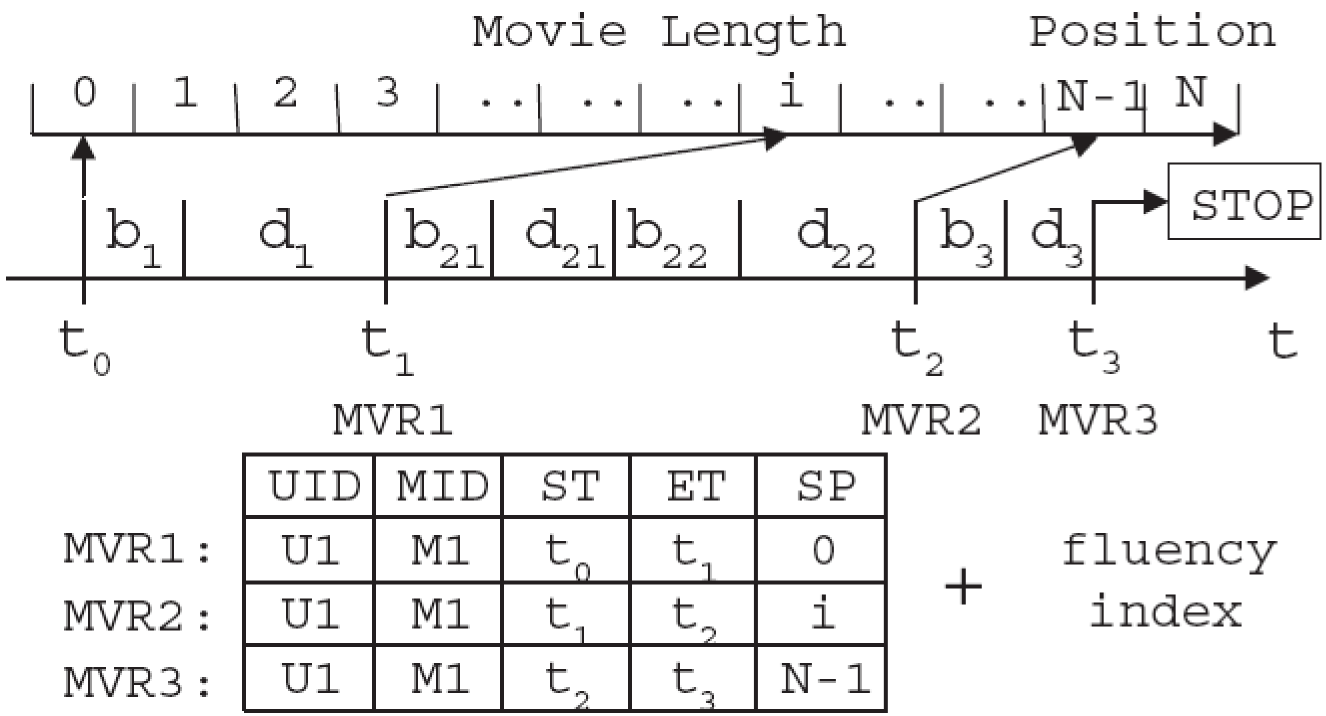
nk i at t / unk i at t

- To provide good scalability and quality viewing, *ATDi*(*t*) has to be greater than 1. In here, *ATDi*(*t*) ≥ 3 for all time *t*.
- 2 peaks for Movie 2 at 12:00 or 19:00.

Figure 12: The ratio of the number of available chunks to the demanded chunks within one day.

User Satisfaction Index (1)

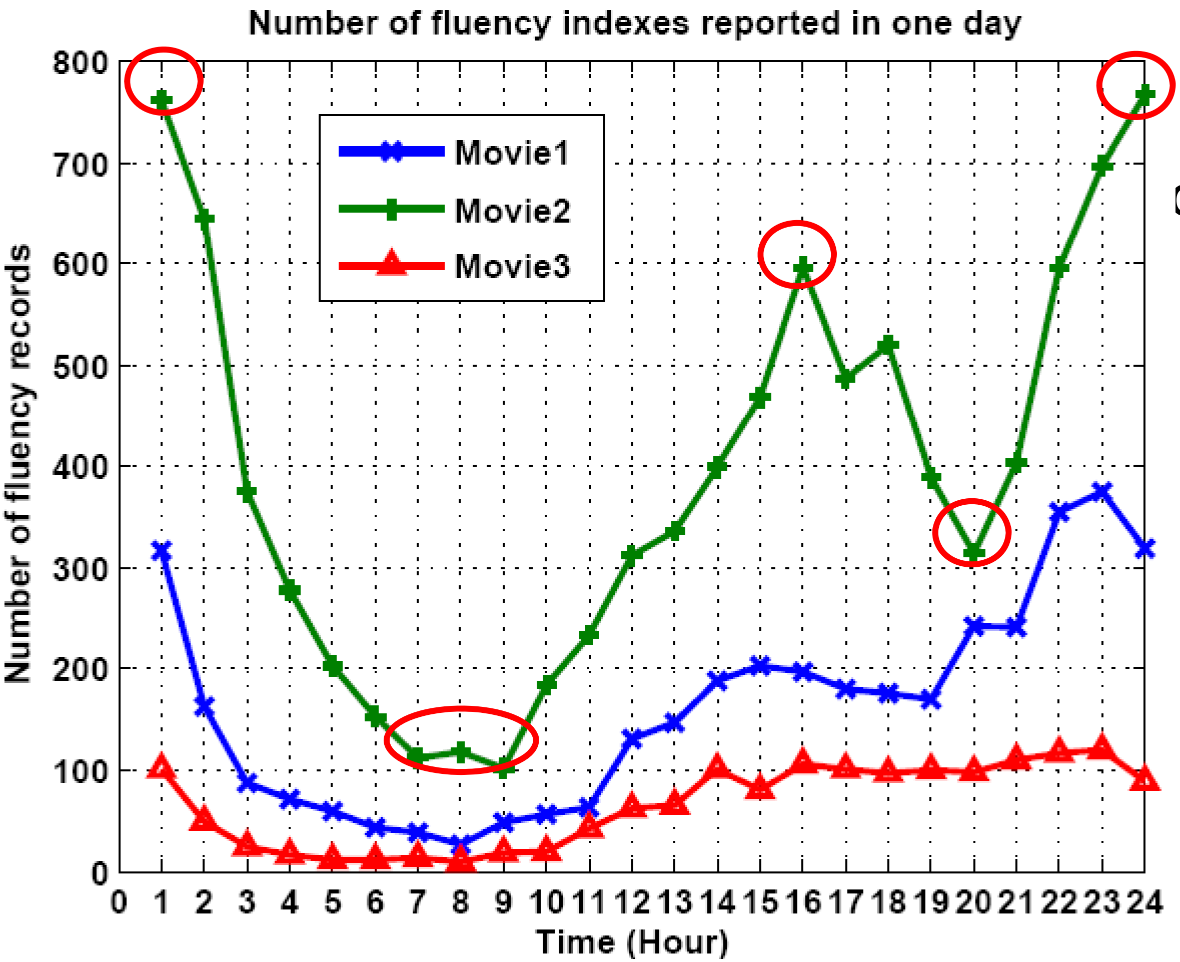
- User satisfaction index is used to measure the
  qu
  - system.
- G
  -
  - y $F(m, i)$



  - The user turns off the P2P-VoD software

User Satisfaction Index (2)

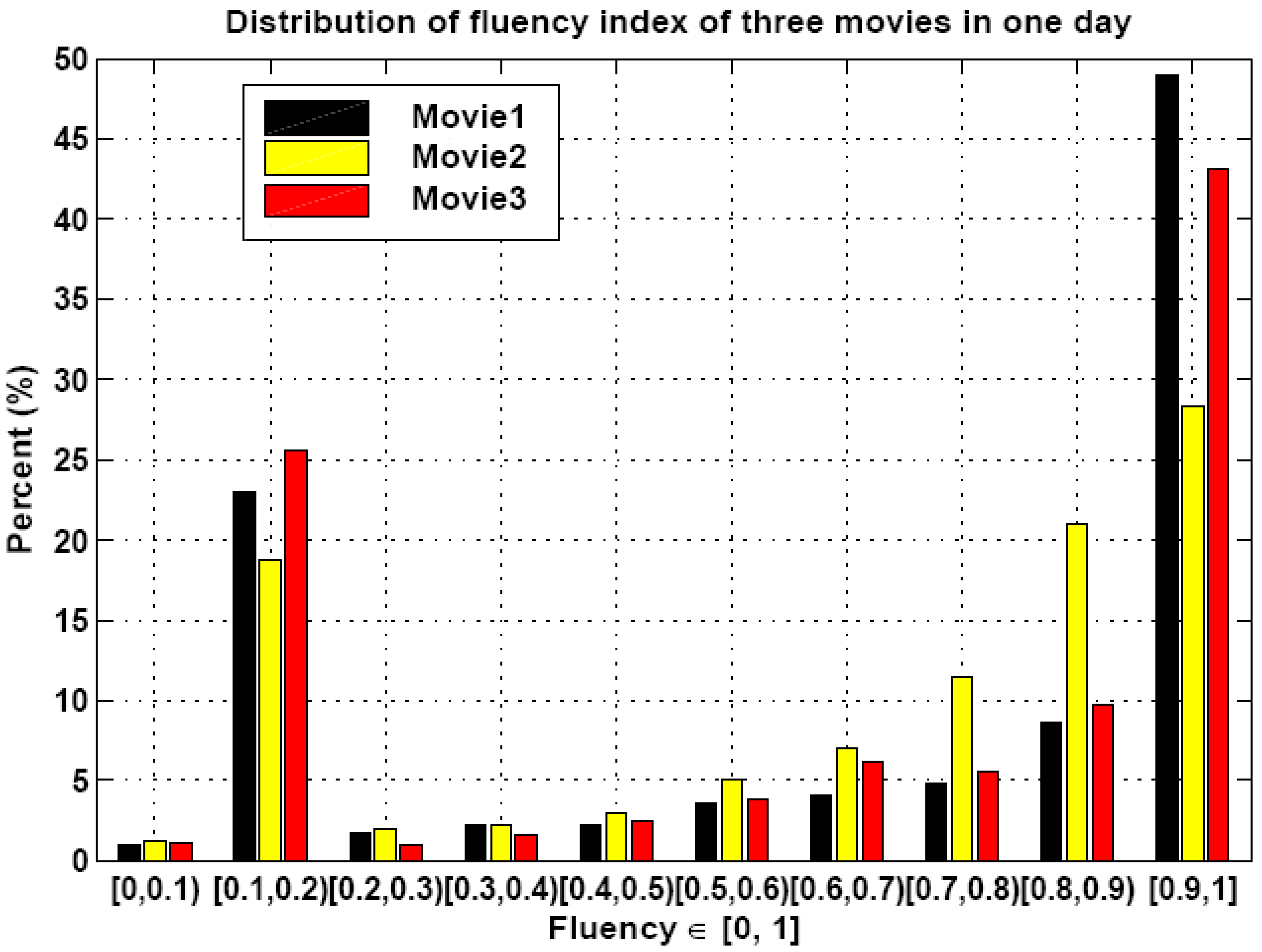
- The nu
  - A go ovie



The number of viewers in the system at different time points.

Figure 15: Number of fluency indexes reported by users to the log server.

User Satisfaction Index (3): The distribution of fluency index



- Good viewing quality: fluency value greater than 0.8
- Poor viewing quality: value less than 0.2
- High percentage of fluency indexes whose values are greater than 0.7.
- Around 20% of the fluency indexes are less than 0.2. There is a high buffering time (which causes long start-up latency) for each viewing operation.

Figure 16: Distribution of fluency index of users within a 24-hour period.

Server Load

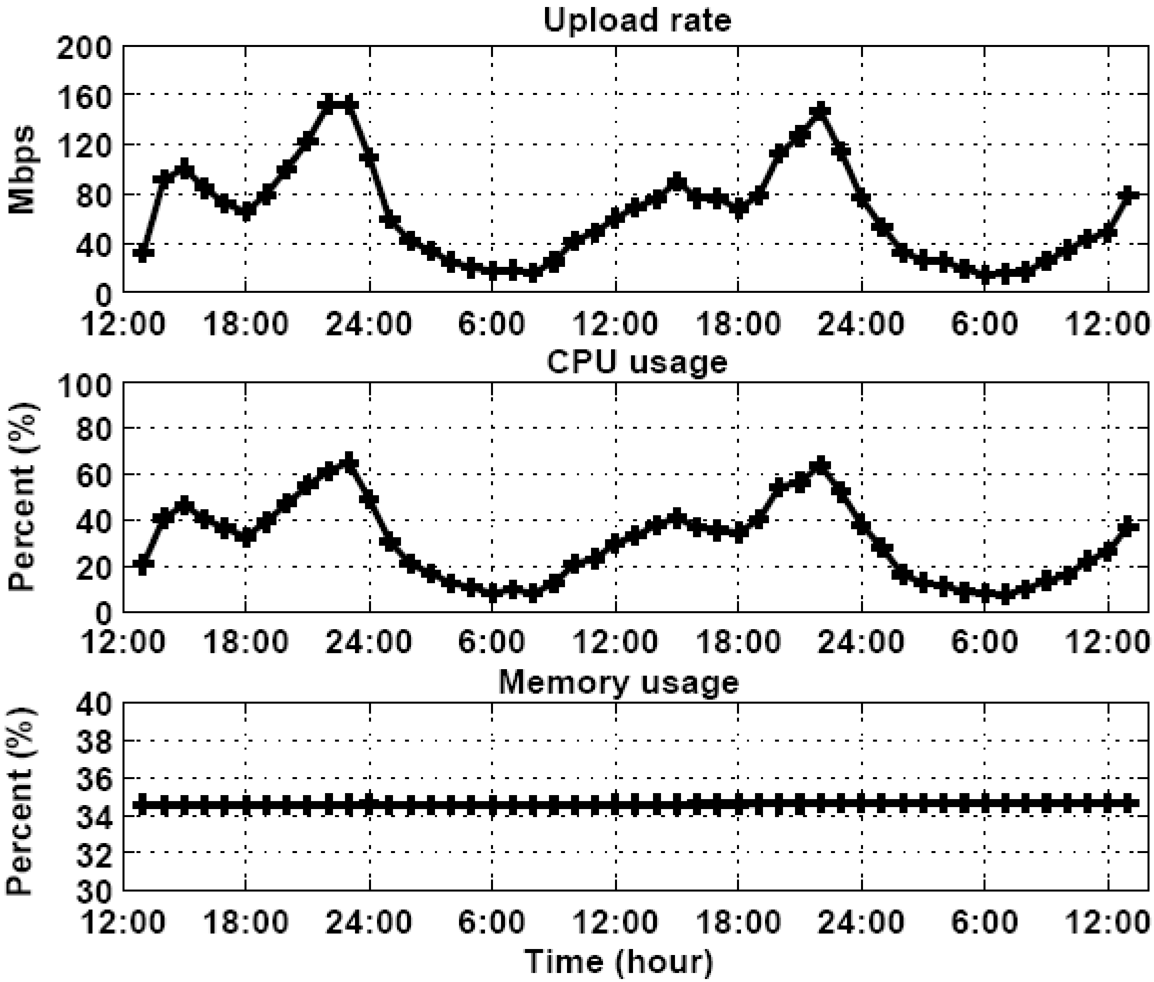


Figure 18: Server load within a 48-hour period.

- The server upload rate and CPU utilization are correlated with the number of users viewing the movies.
- P2P technology helps to reduce the server's load.
- The server has implemented the memory-pool technique which makes the usage of the memory more efficient. (The memory usage is very stable)

Server Load(Cont.)

Upload (Kbps)	# of Peers (%)	Download (Kbps)	# of Peers (%)
$[0, 200)$	65616(35.94%)	$[0, 360)$	46504(25.47%)
$[200, 360)$	51040(27.96%)	$[360, 600)$	118256(64.78%)
$[360, 600)$	45368(24.86%)	$[600, 1000)$	14632(8.01%)
$[600, 1000)$	9392(5.14%)	$[1000, 2000)$	3040(1.67%)
> 1000	11128(6.10%)	> 2000	112(0.07%)
Total	182544	Total	182544

Table 4: Distribution of average upload and download rate in one-day measurement period.

- Measure on May 12, 2008.
- The average rate of a peer downloading from the server is 32Kbps and 352Kbps from the neighbor peers.
- The average upload rate of a peer is about 368Kbps.
- The average server loading during this one-day measurement period is about 8.3%.

NAT Related Statistics

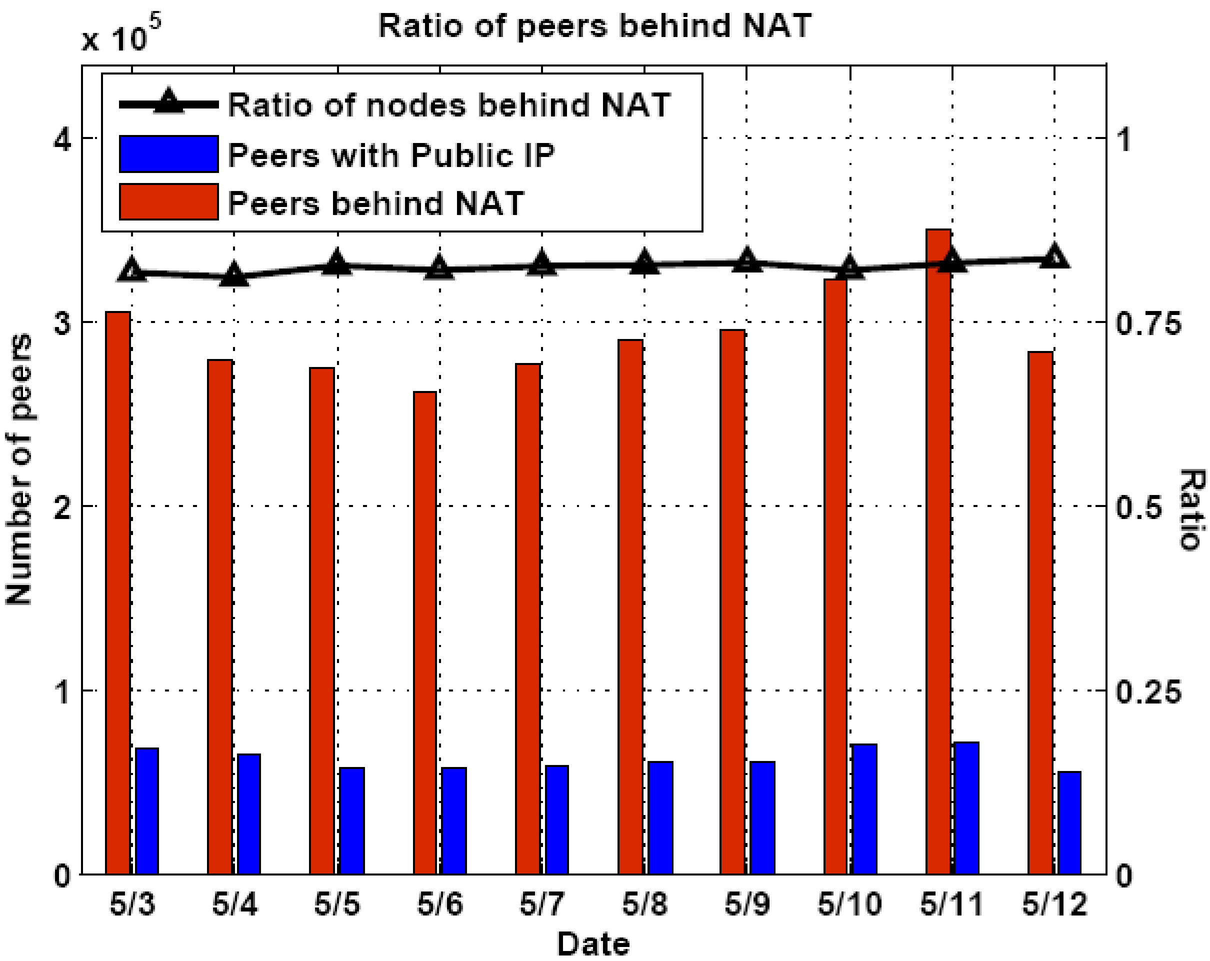


Figure 19: Ratio of peers behind NAT boxes within a 10-day period.

NAT Related Statistics(Cont.)

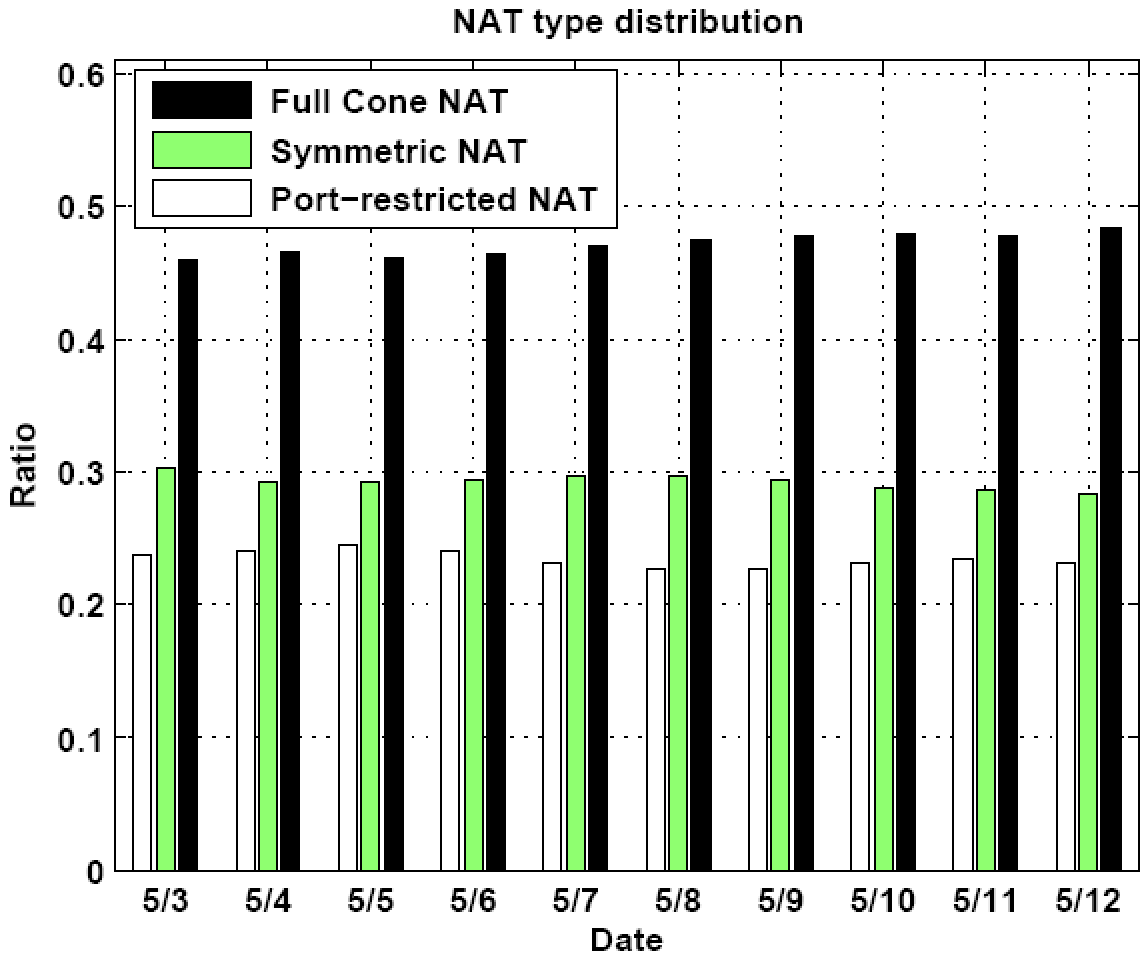


Figure 20: Distribution of peers with different NAT types within a 10-day period.

Conclusions

- We present a general architecture and important building blocks of realizing a P2P-VoD system.
  - Performing dynamic movie replication and scheduling
  - Selection of proper transmission strategy
  - Measuring User satisfaction level
- Our work is the first to conduct an in-depth study on practical design and measurement issues deployed by a real-world P2P-VoD system.
- We have measured and collected data from this real-world P2P-VoD system with totally 2.2 million independent users.

References

- [13] Y. Guo, K. Suh, J. Kurose, and D. Towsley. P2cast: peer-to-peer patching scheme for vod service. In *Proceedings of the 12th ACM International World Wide Web Conference (WWW)*, Budapest, Hungary, May 2003.
- [14] A. A. Hamra, E. W. Biersack, and G. Urvoy-Keller. A pull-based approach for a vod service in p2p networks. In *IEEE HSNMC*, Toulouse, France, July 2004.
- [15] X. Hei, C. Liang, Y. Liu, and K. W. Ross. A measurement study of a large-scale P2P iptv system. *IEEE Transactions on Multimedia*, 9(8):1672–1687, December 2007.
- [16] A. Hu. Video-on-demand broadcasting protocols: a comprehensive study. In *Proceedings of IEEE INFOCOM'01*, Anchorage, AK, USA, April 2001.
- [17] C. Huang, J. Li, and K. W. Ross. Can internet video-on-demand be profitable? In *Proceedings of ACM SIGCOMM'07*, Kyoto, Japan, August 2007.
- [18] R. Kumar, Y. Liu, and K. W. Ross. Stochastic fluid theory for p2p streaming systems. In *Proceedings of IEEE INFOCOM'07*, May 2007.
- [22] Y. Zhou, D. M. Chiu, and J. C. S. Lui. A simple model for analyzing p2p streaming protocols. In *Proceedings of IEEE ICNP'07*, October 2007.

IV. ISP Friendliness in P2P Systems

Outline

H. Xie, et al., “P4P: Provider Portal for P2P Applications”, ACM SIGCOMM 2008.

[Acknowledgement: Slides taken from authors’ Sigcomm presentation]

P2P: Benefits and Challenges

P2P is a key to content delivery

- Low costs to content owners/distributors
- Scalability

Challenge

- Network-obliviousness usually leads to network inefficiency
  - Intradomain: for Verizon network, P2P traffic traverses 1000 miles and 5.5 metro-hops on average
  - Interdomain: 50%-90% of existing local pieces in active users are downloaded externally*

*Karagiannis et al. Should Internet service providers fear peer-assisted content distribution? In Proceeding of IMC 2005

ISP Attempts to Address P2P Issues

- Upgrade infrastructure
- Customer pricing
- Rate limiting, or termination of services

- P2P caching

ISPs cannot effectively address network efficiency alone

Locality-aware P2P: P2P's Attempt to Improve Network Efficiency

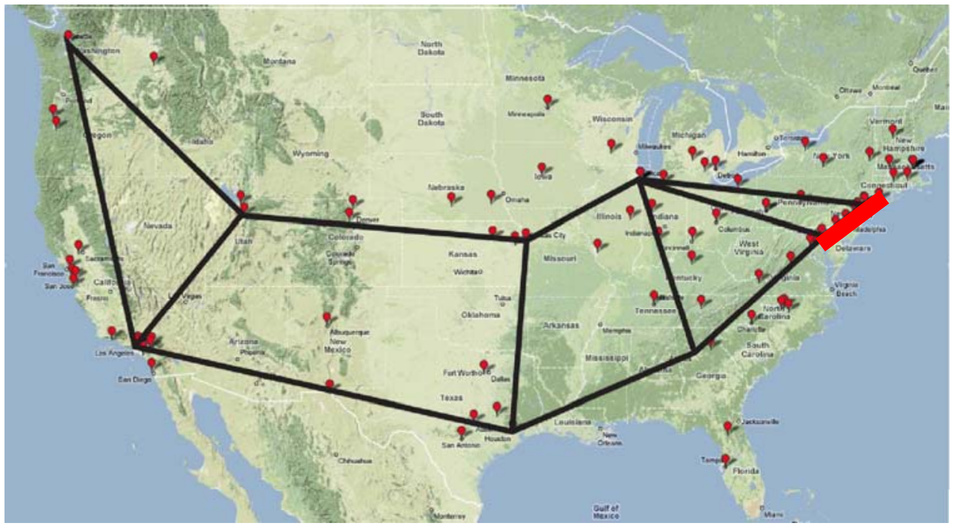
- P2P has flexibility in shaping communication patterns
- Locality-aware P2P tries to use this flexibility to improve network efficiency
  - E.g., Karagiannis et al. 2005, Bindal et al. 2006, Choffnes et al. 2008 (Ono)

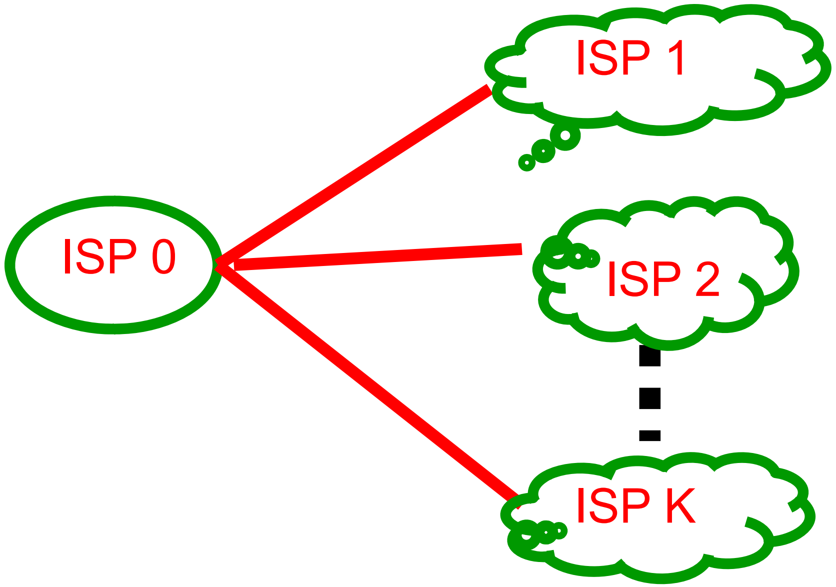
Problems of Locality-aware P2P

- Locality-aware P2P needs to reverse engineer network topology, traffic load and network policy
- Locality-aware P2P may not achieve network efficiency

Choose congested links

Traverse costly interdomain links





A Fundamental Problem

- Feedback from networks is limited
  - E.g., end-to-end flow measurements or limited ICMP feedback

P4P Goal

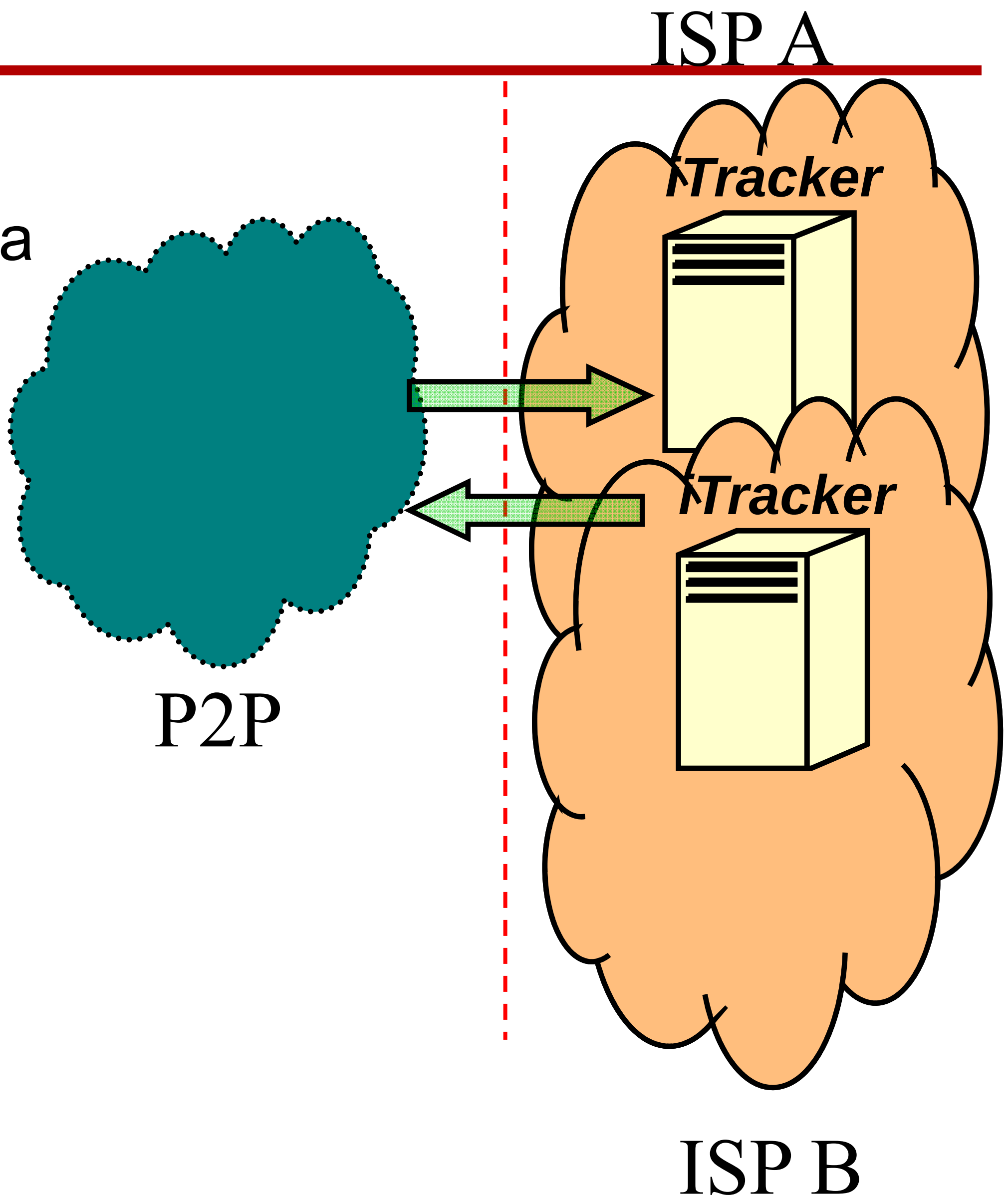
Design a framework to enable better cooperation between networks and P2P

P4P: Provider Portal for (P2P) Applications

P4P Architecture

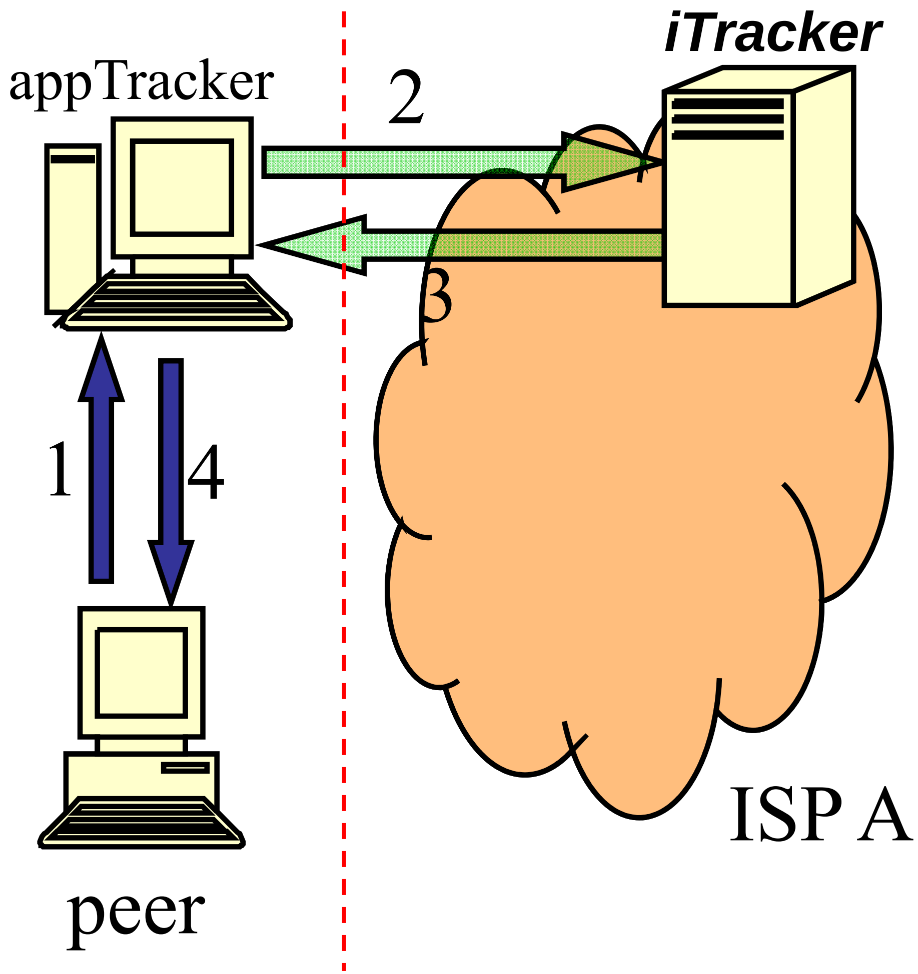
- Providers
  - publish information via iTracker

- Applications
  - query providers' information
  - adjust traffic patterns accordingly



Example:Tracker-based P2P

- Information flow
  - 1. peer queries appTracker
  - 2/3. appTracker queries iTracker
  - 4. appTracker selects a set of active peers

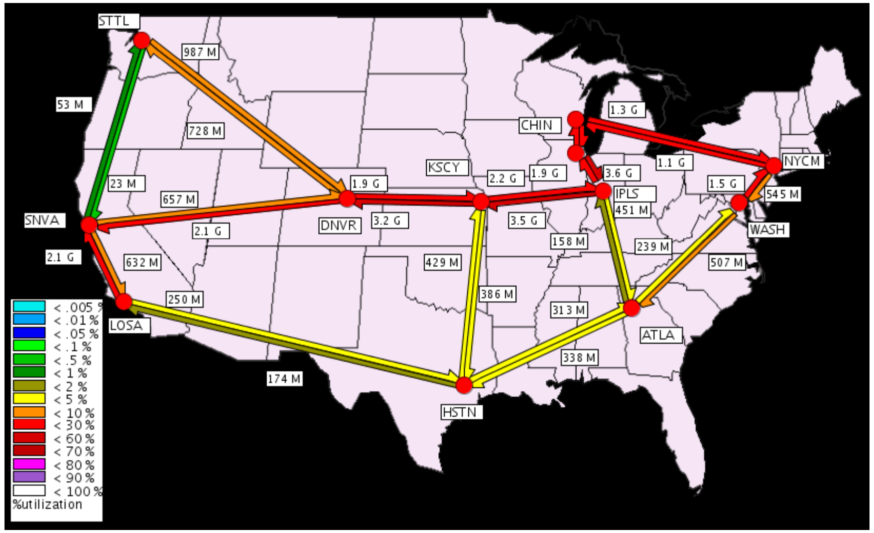


Challenges

- ISPs and applications have their own objectives/constraints
  - ISPs have diverse objectives
  - Applications also have diverse objectives
- Desirable to have
  - Providers: application-agnostic
  - Applications: network-agnostic

A Motivating Example

- ISP objective:
  - Focus on intradomain
  - Minimize maximum link utilization (MLU)



- P2P objective:
  - Optimize completion time

Specifying ISP Objective

❖ ISP Objective

- Minimize MLU

❖ Notations:

- Assume K P2P applications in the ISP's network
- b_e: background traffic volume on link e
- c_e: capacity of link e
- $I_e(i,j)$ = 1 if link e is on the route from i to j
- t^k : a traffic demand matrix $\{t^k_{ij}\}$ for each pair of nodes (i,j)

$$\min \max_{e \in E} \left(b_e + \sum_k \sum_{i \neq j} t^k_{ij} I_e(i,j) \right) / c_e$$

Specifying P2P Objective

- P2P Objective
  - Optimize completion time
- Using a fluid model, we can derive that:
  optimizing P2P completion time

  $\Rightarrow$

  maximizing up/down link capacity usage

$$
\begin{aligned}
&\max \sum_{i} \sum_{j \neq i} t_{ij} \\
&s.t. \forall i, \sum_{j \neq i} t_{ij} \leq u_i \,, \\
&\forall i, \sum_{j \neq i} t_{ji} \leq d_i \,, \\
&\forall i \neq j, t_{ij} \geq 0
\end{aligned}
$$

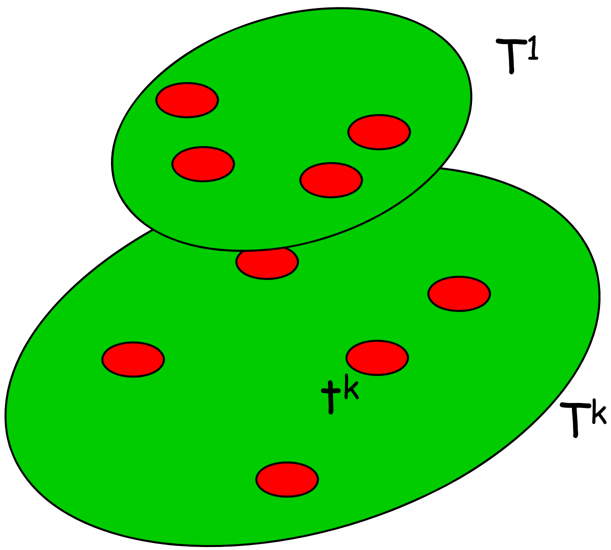
*Modeling and performance analysis of bittorrent-like peer-to-peer networks. Qiu et al. Sigcomm '04

System Formulation

❖ Combine the objectives of provider and application

$$\min \max_{e \in E} (b_e + \sum_k \sum_{i \neq j} t_{ij}^k I_e(i,j)) / c_e$$

s.t., for any k,



$$\max \sum_i \sum_{j \neq i} t_{ij}^k$$

$$s.t. \forall i, \sum_{j \neq i} t_{ij}^k \leq u_i^k,$$

$$\forall i, \sum_{j \neq i} t_{ji}^k \leq d_i^k,$$

$$\forall i \neq j, t_{ij}^k \geq 0$$

Difficulties

- A straightforward approach: centralized solution
  - Applications: ship their information to ISPs
  - ISPs: solve the optimization problem
- Issues
  - Not scalable
  - Not application-agnostic
  - Violation of P2P privacy

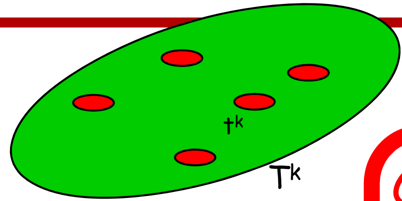
$$\boxed{\min \max_{e \in E} (b_e + \sum_k \sum_{i \neq j} t_{ij}^k I_e(i,j)) / c_e}$$

s.t., for any k,

$$\boxed{\begin{aligned} & \max \sum_i \sum_{j \neq i} t_{ij}^k \\ & s.t. \forall i, \sum_{j \neq i} t_{ij}^k \leq u_i^k, \\ & \forall i, \sum_{j \neq i} t_{ji}^k \leq d_i^k, \\ & \forall i \neq j, t_{ij}^k \geq 0 \end{aligned}}$$

Key Contribution: Decoupling ISP/P2Ps

$$\min_{\forall k: t^k \in T^k} \max_{e \in E} \left(b_e + \sum_k \sum_{i \neq j} t_{ij}^k I_e(i,j)\right) / c_e$$



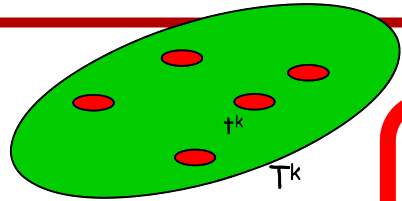
$$\min_{\forall k: t^k \in T^k} \alpha$$

$$s.t. \quad \forall e \quad b_e + \sum_k \sum_{i \neq j} t_{ij}^k I_e(i,j) \leq \alpha c_e$$

Constraints couple ISP/P2Ps together!

Key Contribution: Decoupling ISP/P2Ps

$$\min_{\forall k: t^k \in T^k} \max_{e \in E} \Big(b_e + \sum_k \sum_{i \neq j} t_{ij}^k I_e(i,j)\Big) / c_e$$



Introduce p_e to decouple the constraints

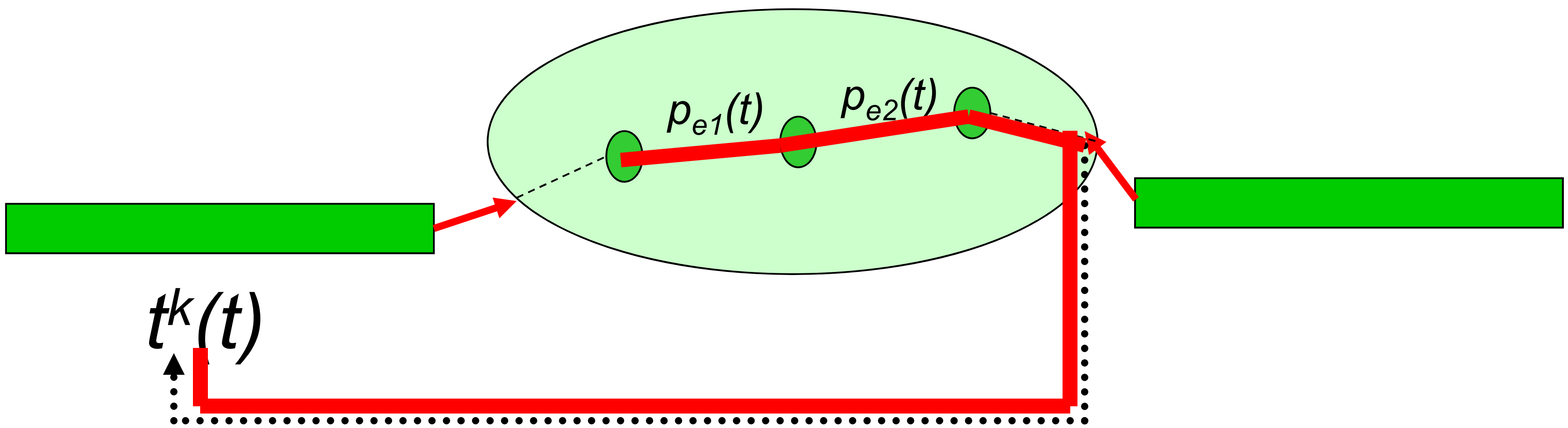
$$\min_{\forall k: t^k \in T^k} \quad \alpha$$

$$s.t. \qquad \forall e: b_e + \sum_k \sum_{i \neq j} t_{ij}^k I_e(i,j) \leq \alpha c_e \qquad p_e$$

$$\max_{\sum p_e c_e = 1} \Big(\sum_e p_e b_e + \sum_k \min_{t^k \in T^k} \sum_{i \neq j} p_{ij} t_{ij}^k\Big)$$

ISP/P2P Interactions

- The interface between applications and providers is $\{p_e\}$
  - Providers: compute $\{p_e\}$, which reflects network status and policy
  - Applications: react and adjust $\{t^k_{ij}\}$ to optimize application objective



Generaliztion

- Generalize to other ISP objectives and P2P objectives

ISPs

- Minimize MLU
- Minimize Bit-Distance Product
- Minimize interdomain cost
- Customized objective

Applications

- Maximize throughput
- Robustness
- Rank peers using p_e
- …

From Optimization Decomposition to Interface Design

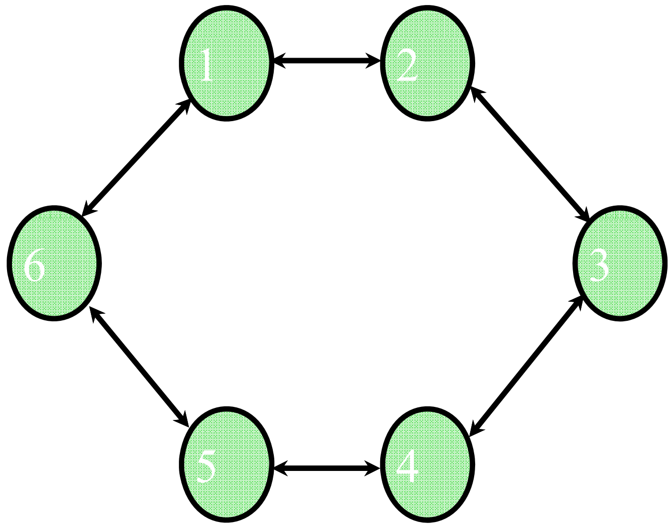
- Issue: scalability

- Technique
  - PIDs: opaque IDs of a group of nodes
    - Clients with the same PID have similar network costs with respect to other clients
  - PID links: network links connecting PIDs (can be "logical" links)
  - p_e: P4P distance for each PID link e

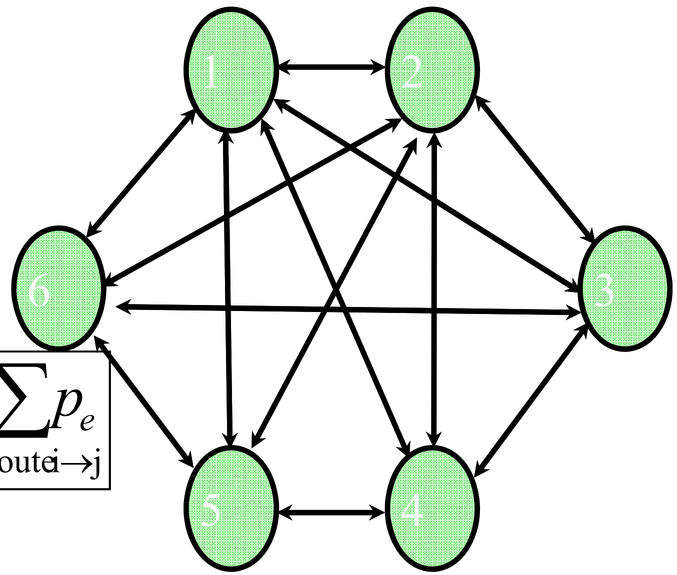
From Optimization Decomposition to Interface Design

- Issue: privacy

- Technique: two views
  - Provider (internal) view

  - Application (external) view
    - pij may be perturbed to preserve privacy

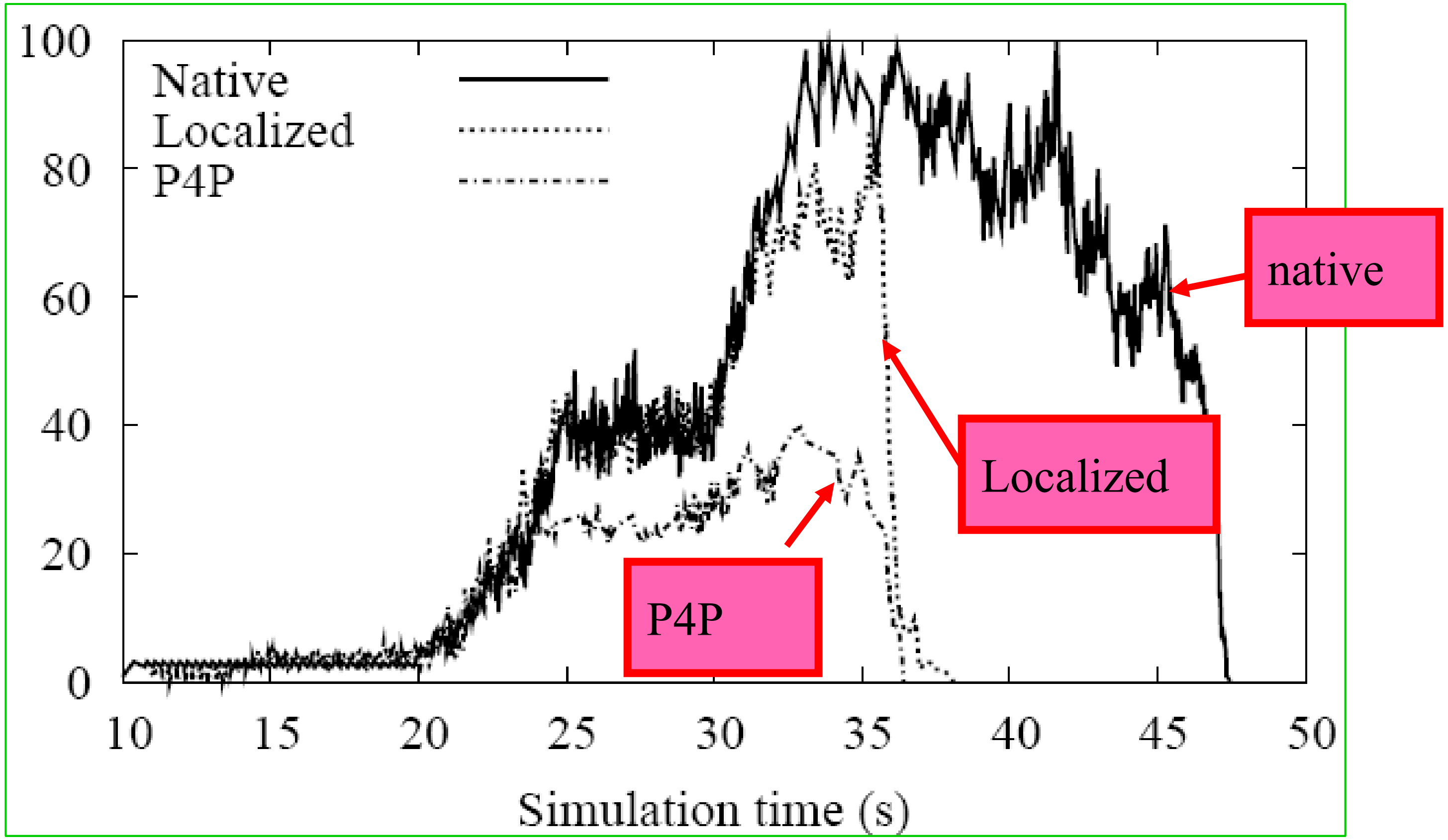


$$p_{ij} = \sum_{e \text{ on route } i \to j} p_e$$

Evaluation Methodology

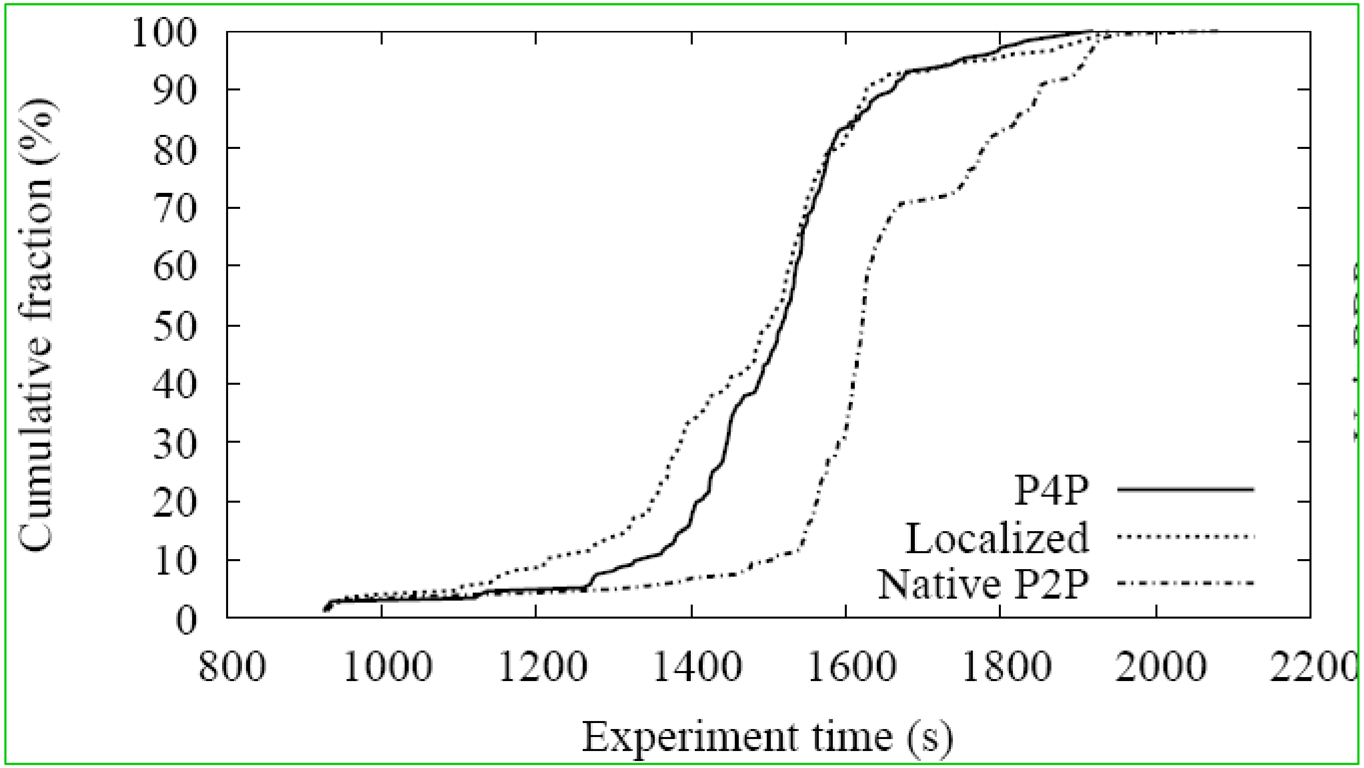
- BitTorrent simulations
  - Build a simulation package for BitTorrent
  - Use topologies of Abilene and Tier-1 ISPs in simulations
- Abilene experiment using BitTorrent
  - Run BitTorrent clients on PlanetLab nodes in Abilene
  - Interdomain emulation
- Field tests using Pando clients
  - Applications: Pando pushed 20 MB video to 1.25 million clients
  - Providers: Verizon and Telefonica provided network topologies

BitTorrent Simulation: Bottleneck Link Utilization



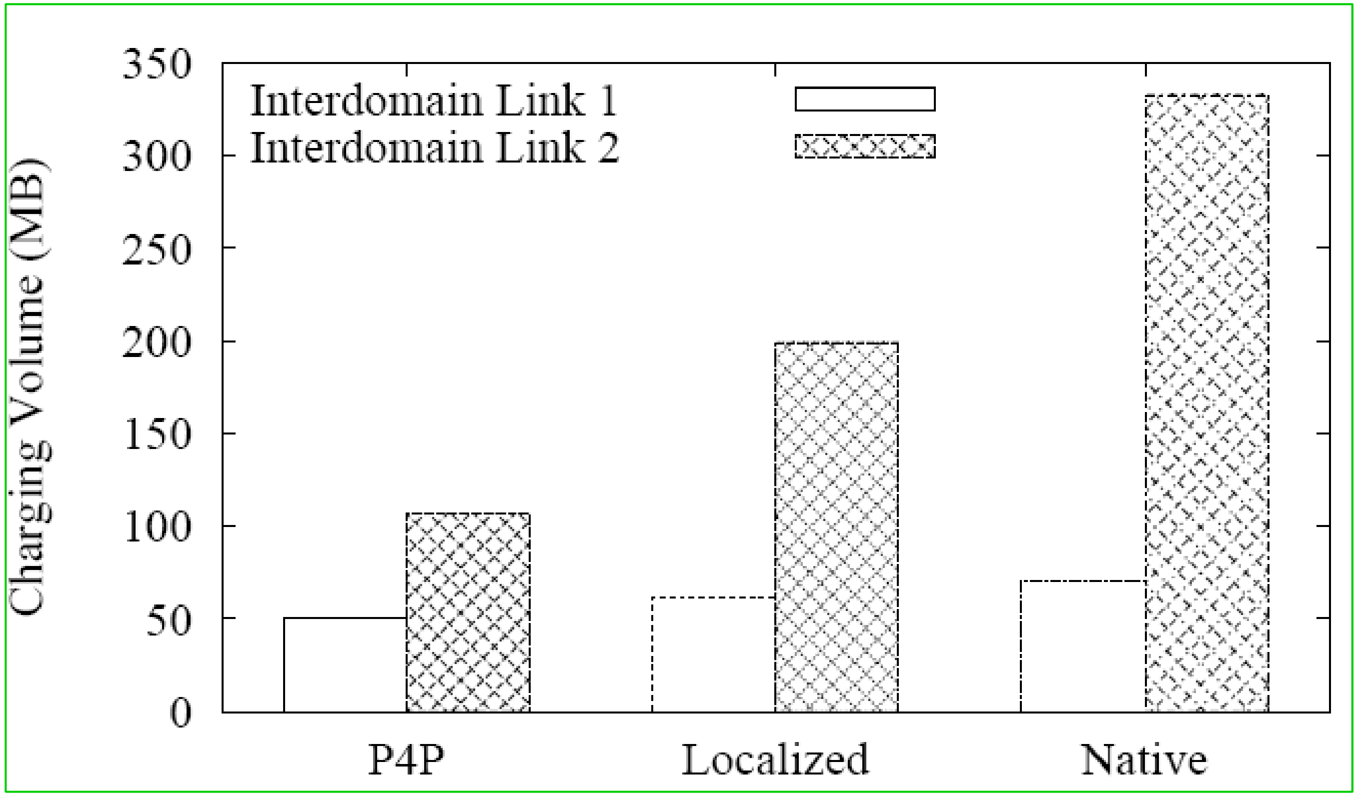
P4P results in less than half utilization on bottleneck links

Abilene Experiment: Completion Time



- P4P achieves similar performance with localized at percentile higher from 50%.
- P4P has a shorter tail.

Abilene Experiment: Charging Volume

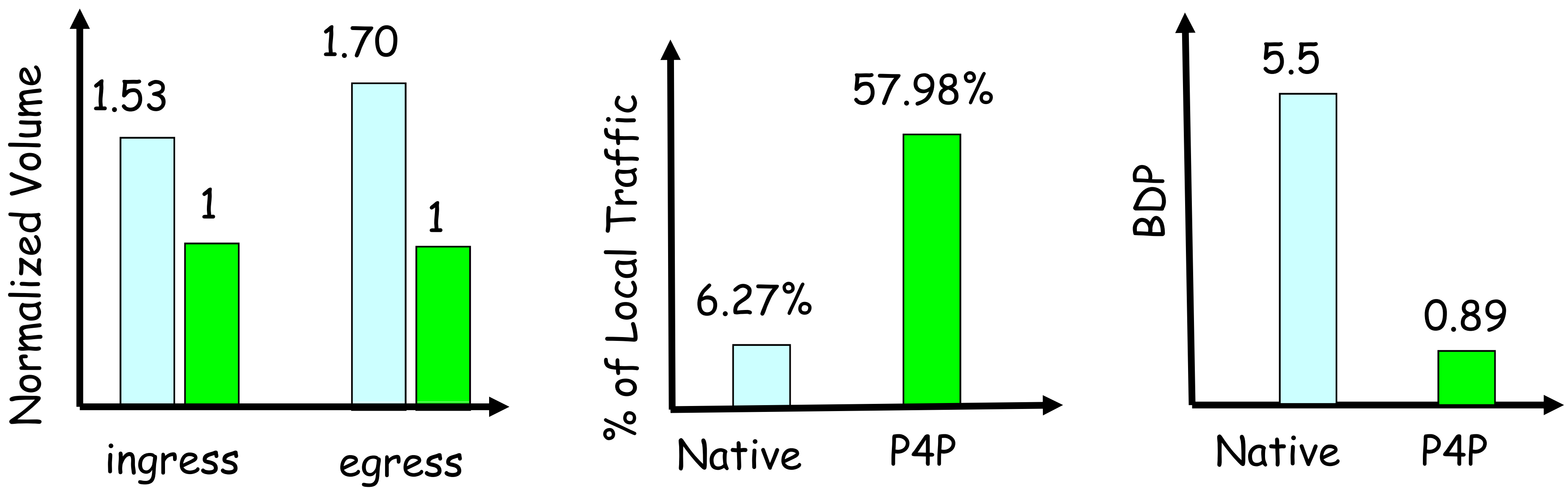


Charging volume of the second link: native BT is 4x of P4P; localized BT is 2x of P4P

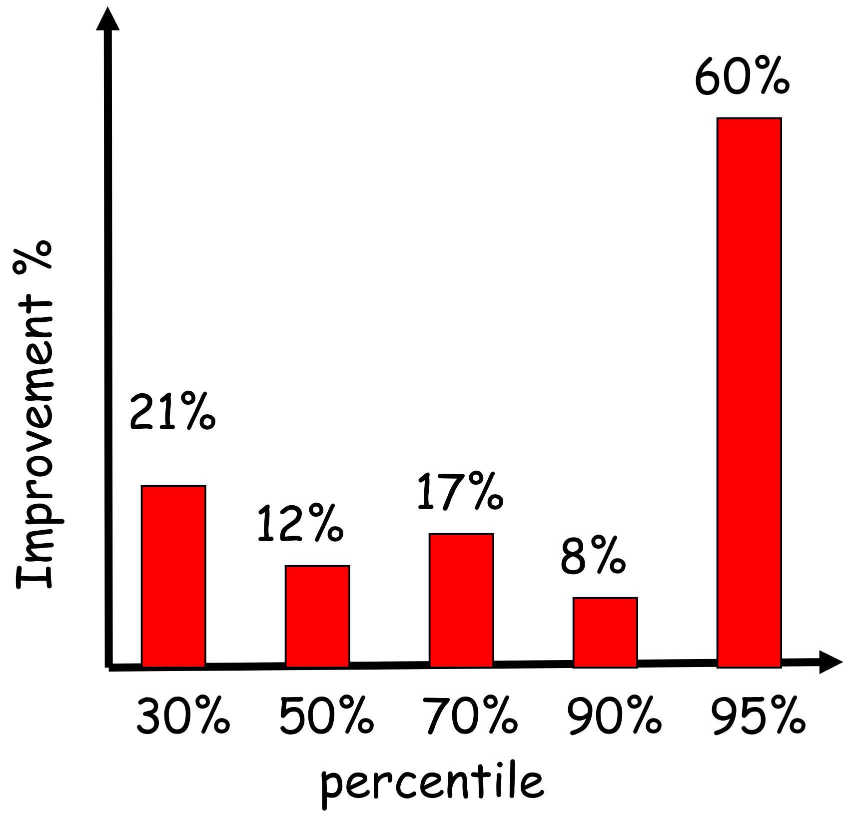
Field Tests: ISP Perspectives

- Interdomain Traffic Statistics
  - Ingress: Native is 53% higher
  - Egress: Native is 70% higher

- Intradomain Traffic Statistics



Field Tests: P2P Completion Time



	Native	P4P	Improvement
30%	243	192	21%
50%	421	372	12%
70%	1254	1036	17%
90%	7187	6606	8%
95%	35046	14093	60%

All P2P clients: P4P improves avg completion time by 23%
FTTH clients: P4P improves avg completion time by 68%

Summary & Future Work

Summary

- Propose P4P for cooperative Internet traffic control
- Apply optimization decomposition to design an extensible and scalable framework
- Concurrent efforts: e.g, Feldmann et al, Telefonica/Thompson

Future work

- P4P capability interface (caching, CoS)
- Further ISP and application integration
- Incentives, privacy, and security analysis of P4P

Backup Slides on P4P Optimization Decomposition

Compute pDistance

- Introducing dual variable p_e (≥ 0) for the inequality of each link e, the dual is

$$\boxed{D(\{p_e\}) = \min_{\alpha; \forall k: t^k \in T^k} \alpha + \sum_e p_e (b_e + \sum_k t_e^k - \alpha c_e)}$$

- To make the dual finite, we need $\sum_e p_e c_e = 1$

- The dual becomes $D(\{p_e\}) = \sum_e p_e b_e + \sum_k \min_{t^k \in T^k} \sum_{i \neq j} p_{ij} t_{ij}^k$
  - p_{ij} is the sum of p_e along the path from PID i to PID j

Update pDistance

- At update m+1,
  - calculate new "shadow prices" for all links,
  - then compute pDistance for all PID pairs

$$p_e(m+1) = [p_e(m+1) + \mu(m)\xi(m)]_S^+$$

μ : step size

ξ : supergradient of $\partial D(\{p_e\})$

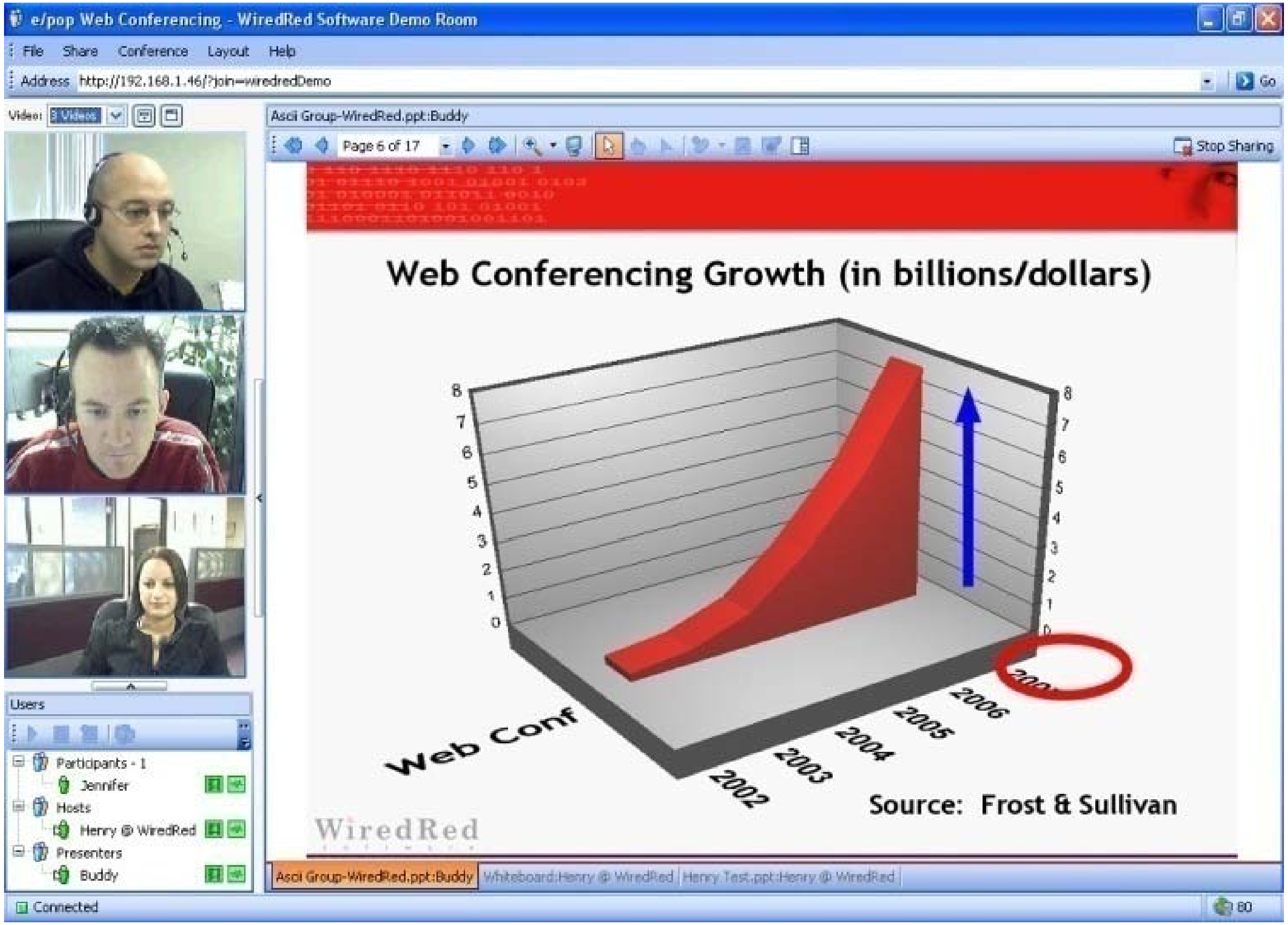
$[]_S^+$: projection to set S

$S : \{p_e : \sum_e p_e c_e = 1; p_e \geq 0\}$

PROPOSITION 1. *Let* $S = \{p | \sum_{e \in E} c_e p_e = 1; \forall e \in E, p_e \geq 0\}$ *and* $p \in S$. *Suppose that* $\{\bar{p}\} \in S$ *is given and that* $\{\bar{t}^k\}$ *is an optimal solution of* $D(\{\bar{p}_e\})$. *Then* $\{\xi | \xi_e = b_e + \sum_k \bar{t}_e^k - \alpha c_e\}$ *is a supergradient of* $D(\cdot)$ *at* $\{\bar{p}_e\}$.

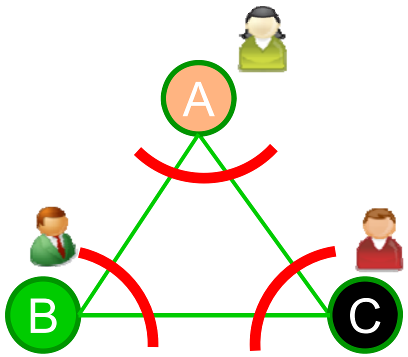
V. P2P Utility Maximization and Its Application in P2P Conferencing

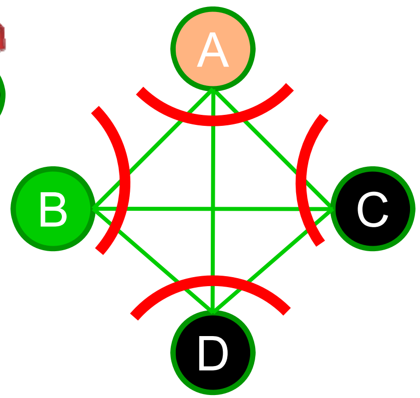
Web Conferencing Application



Multi-party Conferencing Scenario

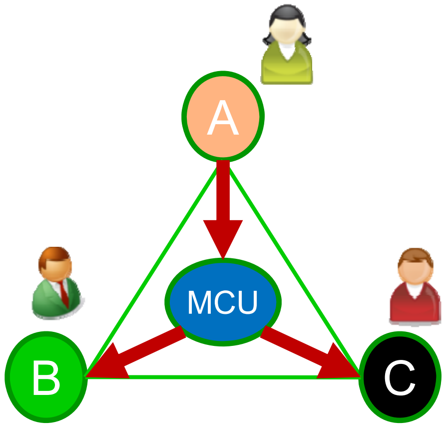
- Every user wants to view audio/video from all other users and is a source of its own audio/video stream
- Maximize Quality-of-Experience (QoE)
- Challenges
  - Network bandwidth limited
  - Require low end-to-end delay
  - Network conditions time-varying
  - Distributed solution not requiring global network knowledge
- Existing Products
  - Apple iChat AV, Skype, Yahoo! Messenger, SightSpeed, hp Halo, Cisco TelePresence, Windows Live Messenger , MS Live Meeting





Comparison of Distribution Approaches

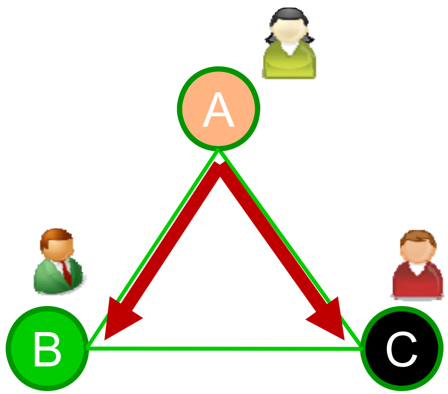
MCU-assisted multicast



High load on MCU, expensive, not scalable with increasing number of peers or groups

Halo

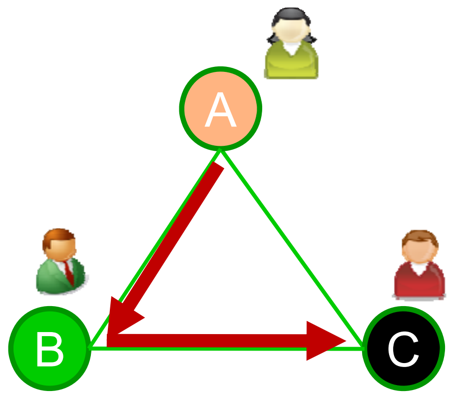
Simulcast



As group size and heterogeneity increases, video quality deteriorates due to peer uplink bandwidth constraint

Apple iChat AV

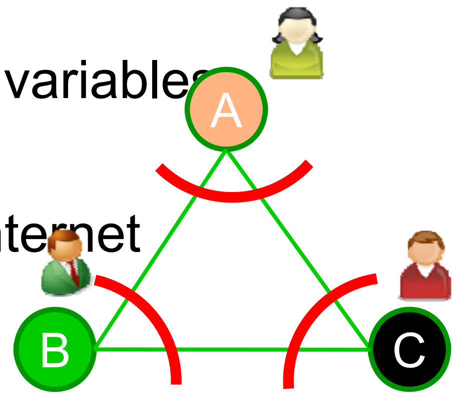
Peer-assisted multicast



Optimal utilization of each peer’s uplink bandwidth, no MCU required but can assist as helper

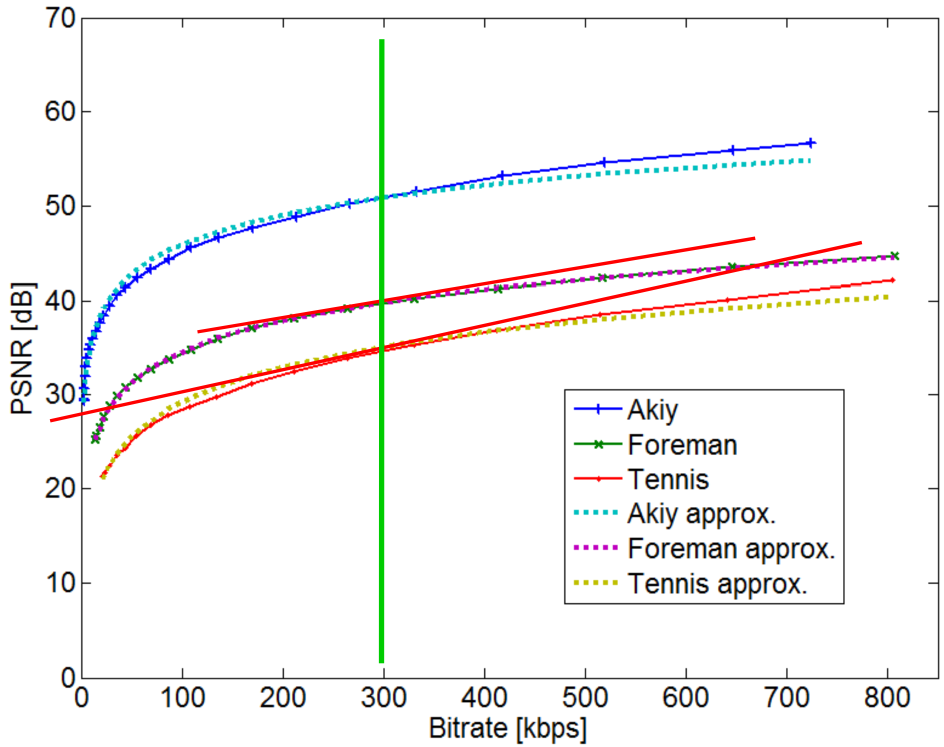
Problem Formulation

- Source s transmitting at rate z_s to all its receivers
- $U_s(z_s)$: (concave) utility associated with video stream of source s
  - Example: PSNR curve
- Only uplinks of peers are bottleneck links
- Maximize total utility of all receivers subject to peer uplink constraints
  - Joint rate allocation and routing problem
  - Linear constraints through introduction of routing variables
  - Concave optimization problem
  - Need distributed solution for deployment in the Internet



Logarithmic Modeling for Utility (PSNR)

- Utility of one peer node defined as $U_s(z_s) = \beta_s \log(z_s)$ **strictly concave**
- Large amount of motion ➔ large β_s
- Peers' utility might change from time to time as they speak/move…



Convex Optimization Problem

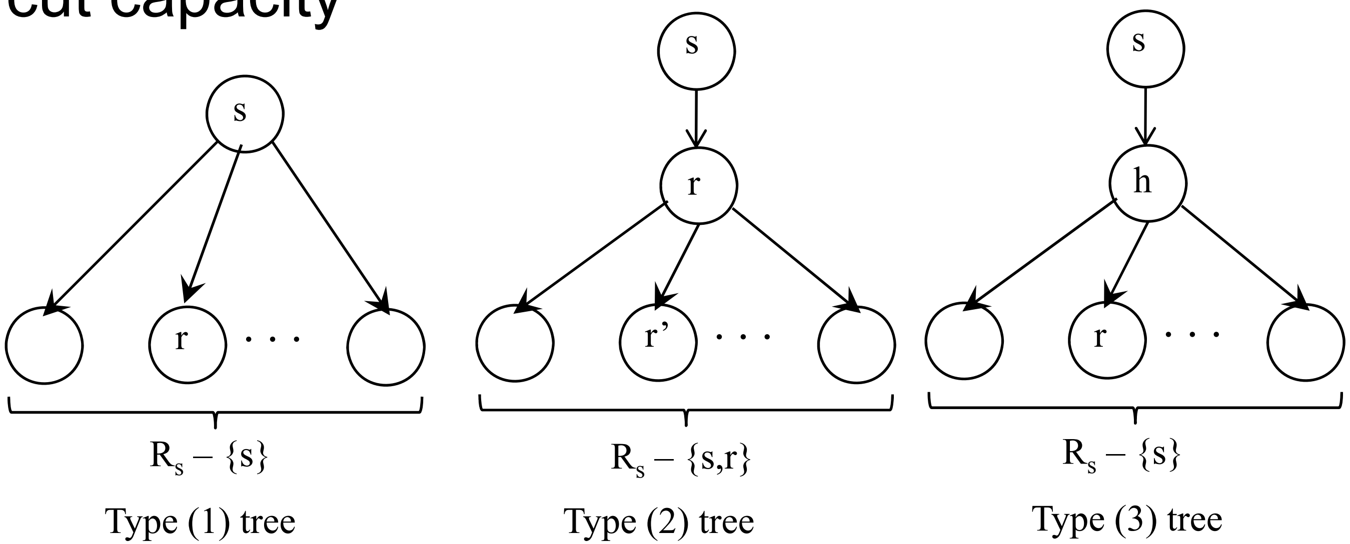
$$\max_z \quad \sum_{s \in S} |R_s| U_s(z_s)$$

$$\text{s.t.} \quad \text{the achievable set of } z$$

- S: set of sources
- R_s : set of receivers for source s
- What is the feasible region for rates $\{z_s\}$?
  - Only peer uplink capacities are bottleneck
  - Allow intra-source or inter-source network coding ?

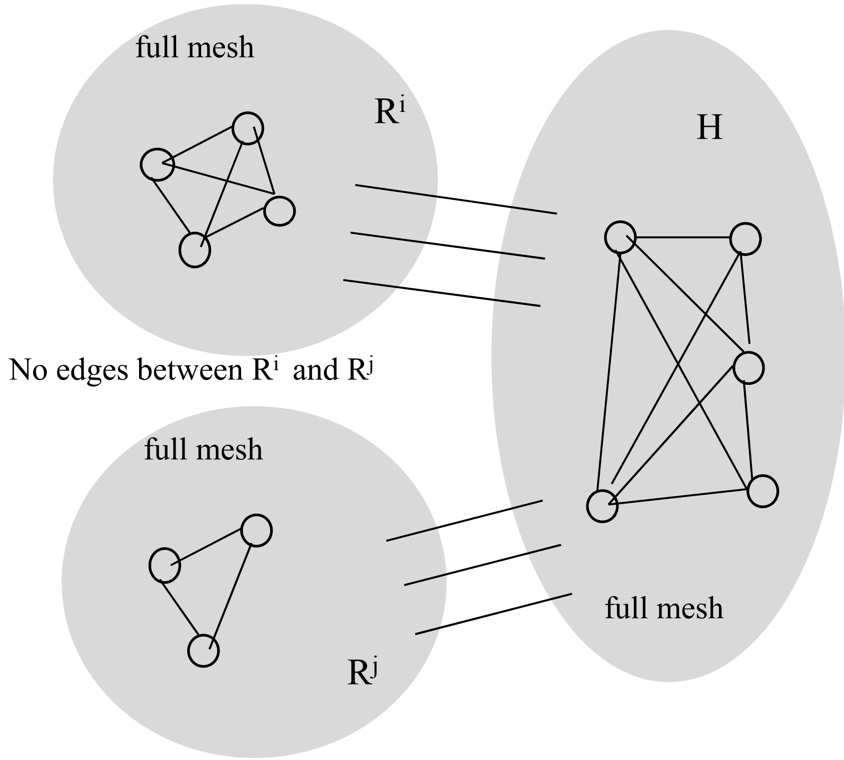
Rate region with Network Coding

- Arbitrary link capacities
  - Routing $\subseteq$ Intra-source coding $\subseteq$ Inter-source coding
- Node uplink capacities only, single source
  - Mutualcast Theorem [Li-Chou-Zhang 05]
  - Routing along linear number of trees achieves min-cut capacity



Rate region with Network Coding …

- Node uplink capacities only, multiple sources
  - No inter-source coding: Linear number of Mutualcast trees per source achieve rate region [Sengupta-Chen-Chou-Li 08]
  - Allow inter-source coding: Linear number of Mutualcast trees per source achieve rate region [Sengupta-Chen-Chou-Li 08] (some restriction on structure of receiver sets)



New Tree-rate Based Formulation

$$\max_x \quad \sum_{s \in S} |R_s| U^s \left(\sum_{m \in s} x_m \right)$$

$$\text{s.t.} \quad y_j \leq C_j, \quad j \in J$$

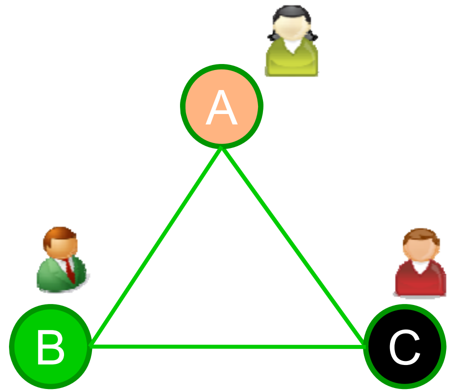
- (Non-strictly) Convex optimization problem with linear constraints
  - y_j : Uplink usage of peer j
  - x_m ($m \in s$): Rate on tree m of source s
  - C_j : Uplink capacity of peer j

Related Work

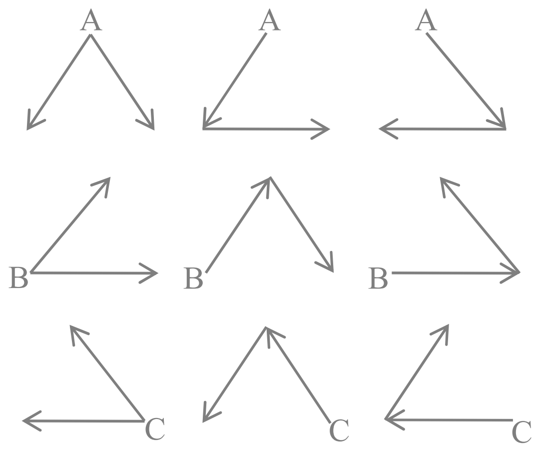
- Utility maximization framework for single-path multicast without network coding [Kelly-Maullo-Tan 98]
- Extensions (without network coding)
  - Multi-path unicast [Han et al 06, Lin-Shroff 06, Voice 06]
  - Single-tree multicast [Kar et al 01]
- Extensions (with single-source network coding)
  - Multicast [Lun et al 06, Wu-Chiang-Kung 06, Chen et al 07]
- What we cover here
  - P2P multicast with multi-source network coding

Need Distributed Rate Control Algorithm

- Best possible rate region achieved by depth-1 and depth-2 trees
  - Determine rate z_s for each source s
  - Determine rates x_m for each source (how much to send on each tree)
- Global knowledge of network conditions or per-source utility functions *should not be required*
  - Adapt to uplink cross-traffic
  - Adapt to changes in utility function (user moving or still)



3 peers



9 multicast trees

Packet Marking Based Primal Algorithm

- Capacity constraint relaxed and added as penalty function to objective

$$\max_{\{x_m\}} \sum_{s\in S} |R_s| U_s(z_s) - \sum_{h\in H} G_h(y_h) - \sum_{j\in J} \int_0^{y_j} q_j(w)\, dw$$

- $q_j(w) = \frac{(w - C_j)^+}{w}$ (packet loss rate or ECN marking probability)
- Simple gradient descent algorithm

$$\dot{x}_m = f_m(x_m) \left(|R_s| U_s'(z_s) - \sum_{h\in m} b_h^m G_h'(y_h) - \sum_{j\in m} b_j^m q_j(y_j) \right)$$

- Global exponential convergence

Queueing Delay Based Primal-Dual Algorithm

- Lagrangian multipliers p_j for each uplink j

$$L(x,p) = \sum_{s \in S} |R_s| U_s(z_s) - \sum_{j \in J} p_j (y_j - C_j)$$

- Primal-dual algorithm

$$\dot{x}_m = k_m \left(U_s'(z_s) - \frac{1}{|R_s|} \sum_{j \in m} b_j^m p_j \right)$$

$$\dot{p}_j = \frac{1}{C_j} (y_j - C_j)_{p_j}^{+},$$

- p_j can be interpreted as queueing delay on peer uplink j
- The term $\frac{1}{|R_s|} \sum_{j \in m} b_j^m p_j$ can be interpreted as average queueing delay of a branch on tree m

Convergence behavior of Primal-Dual algorithm

- There exist cases where primal-dual system does not converge in multi-path setting [Voice 06]
- Positive Results [Chen-Ponec-Sengupta-Li-Chou 08]
  - For P2P multi-party conferencing, all (x,p) trajectories of the system converge to one of its equilibria if for source s, all its k_m ($m \in s$) take the same value
  - For P2P content dissemination , all (x,p) trajectories of the system converge to one of its equilibria if a mild condition (involving k_m and C_j) is satisfied

Convergence behavior of Primal-Dual algorithm

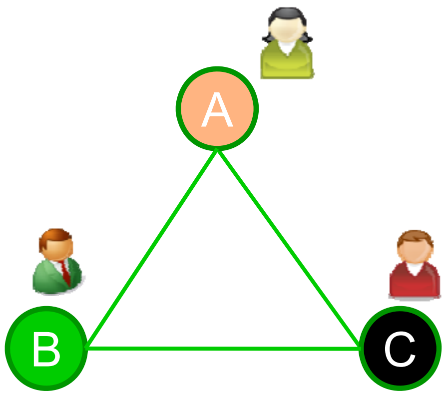
- Trajectories of the system converge to an invariant set, which contains equilibria and limit cycles
  - On the invariant set, the non-linear system reduces to a marginally stable linear system
- Trajectories of the system converge to its equilibria if p is completely observable through [z, y^H] in the reduced linear system
- Mild condition for P2P dissemination scenario
  - *For all* $1 \le i \ne j \le n$, $\xi_i \ne \xi_j$, *where*

$$\xi_l = \begin{cases} \frac{(n_l - 1)n_l}{C_l} k_{ll}, & 1 \le l \le n_s; \\ \frac{1}{C_l} \sum_{j: l \in R_j} (n_j - 1)^2 k_{jl}, & \textit{otherwise} \end{cases}$$

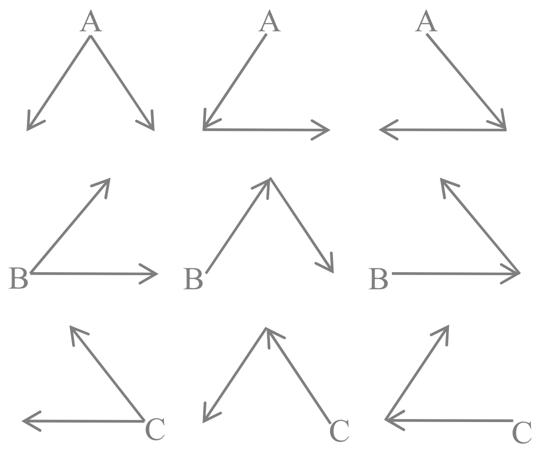
  - $k_{ii} < \frac{C_i}{2C_j} k_{ij}$, *for all* $1 \le i \le n_s$ *and* $n_s < j \le n$.

Implementation of Primal-Dual Algorithm

- What each peer node does?
  - *Sending* its video through trees for which it is a root
  - *Adapting sending rates*
  - *Forwarding* video packets of other peers
  - *Estimating* queuing delay



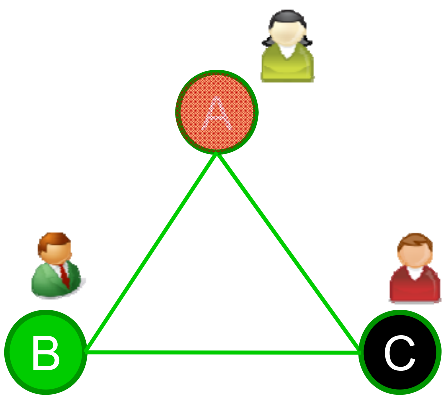
3 peers



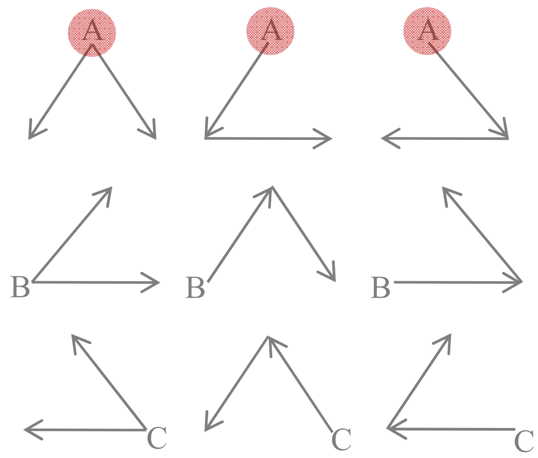
9 multicast trees

Implementation Details

- What each peer node does?
  - *Sending* its video through trees for which it is a root
  - *Adapting sending rates*
  - *Forwarding* video packets of other peers
  - *Estimating* queuing delay



3 peers

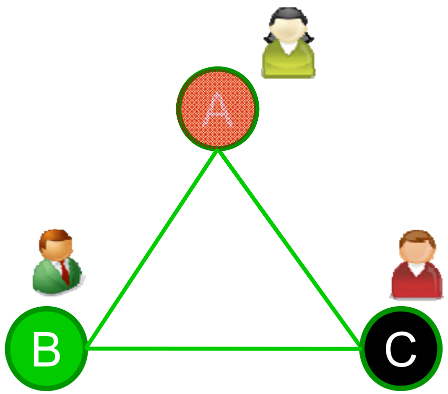


9 multicast trees

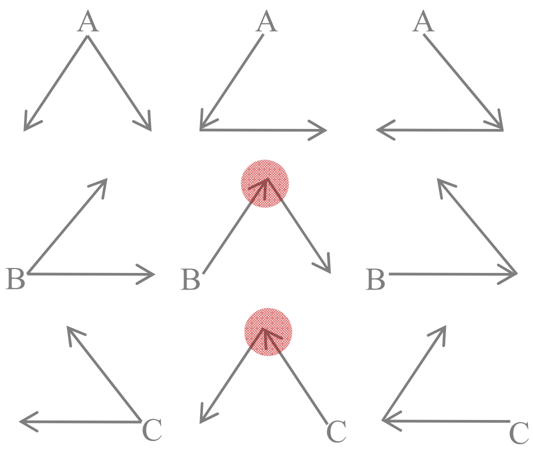
Implementation Details

- What each peer node does?
  - *Sending* its video through trees for which it is a root
  - *Adapting sending rates*
  - *Forwarding* video packets of other peers
  - *Estimating* queuing delay

Helper's functionality

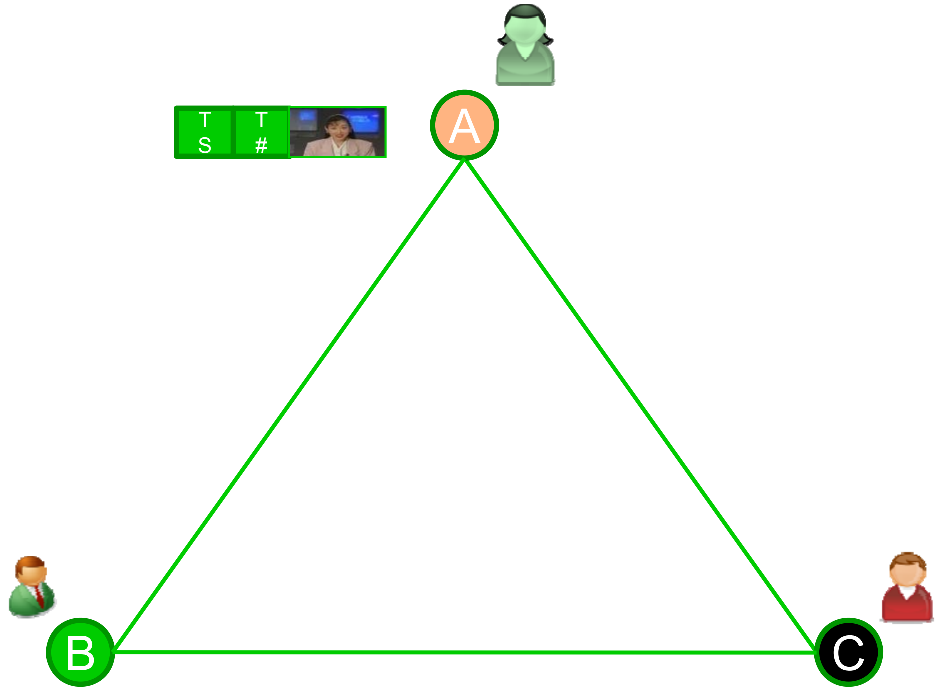


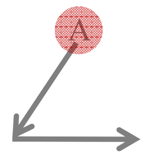
3 peers



9 multicast trees

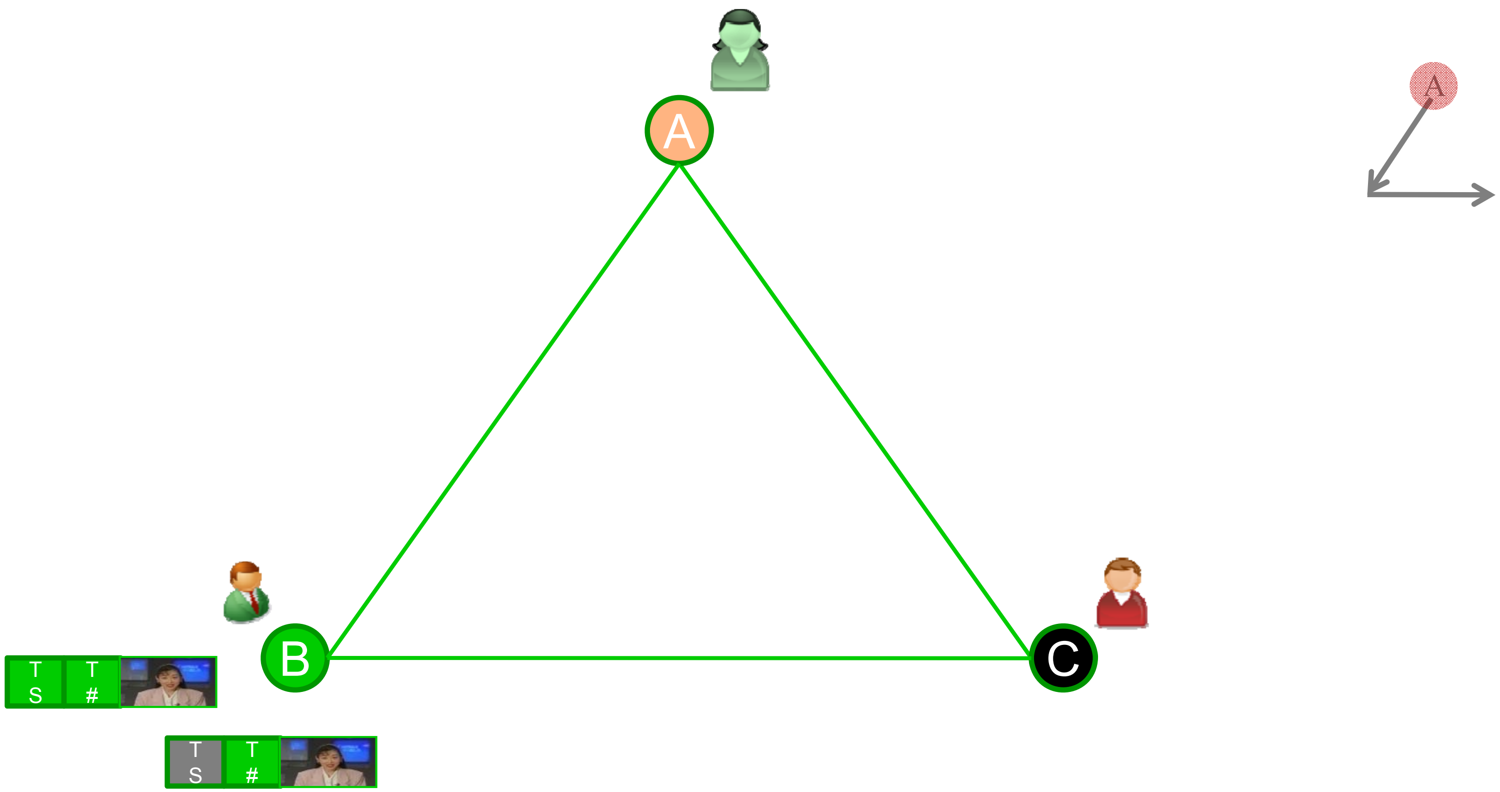
Sending & Forwarding Video



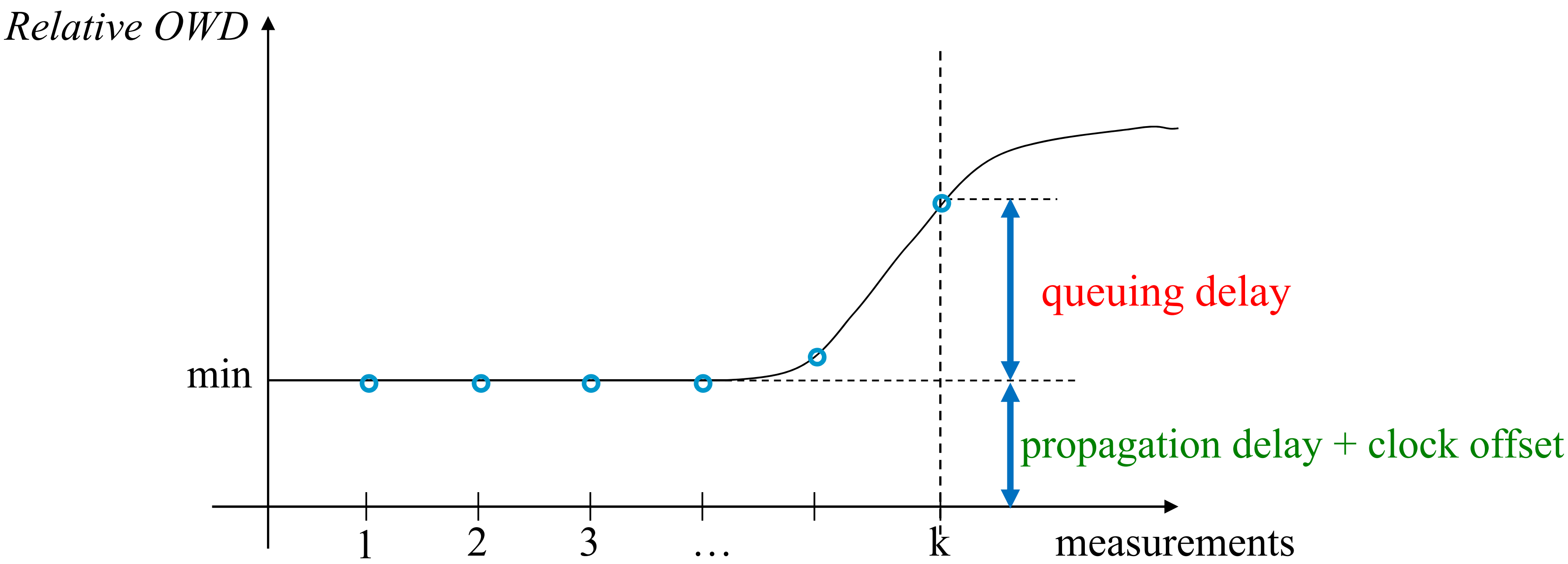


Each packet contains a ***timestamp*** and a ***tree number***

Sending & Forwarding Video



Estimating Queuing Delay Based on Relative *One Way Delay* (OWD) Measurements

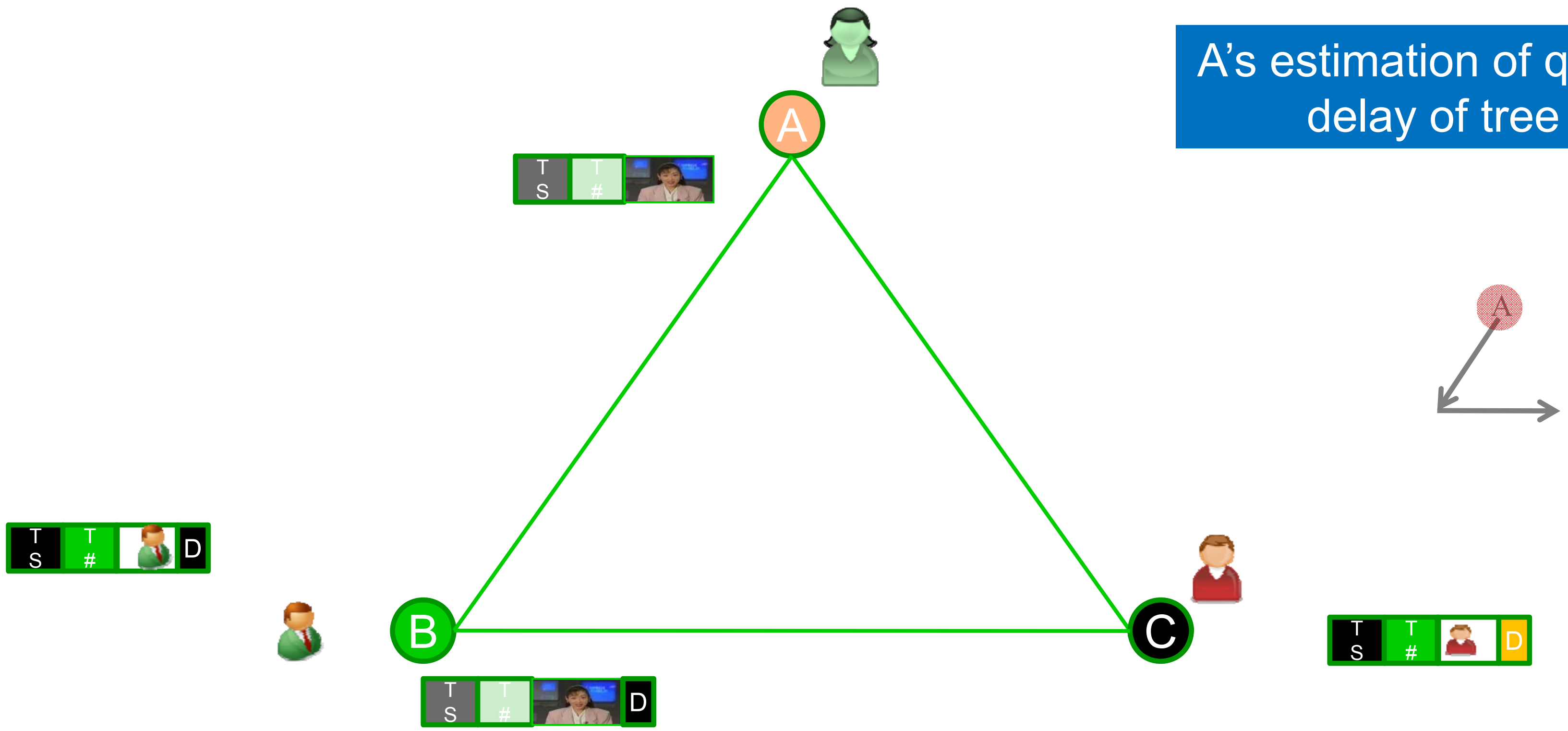


Relative OWD = propagation delay (constant) + clock offset (constant)
+ queuing delay (variable)

No clock synchronization across peers

Queuing delay information piggybacked to video packets

A's estimation of queuing delay of tree 2



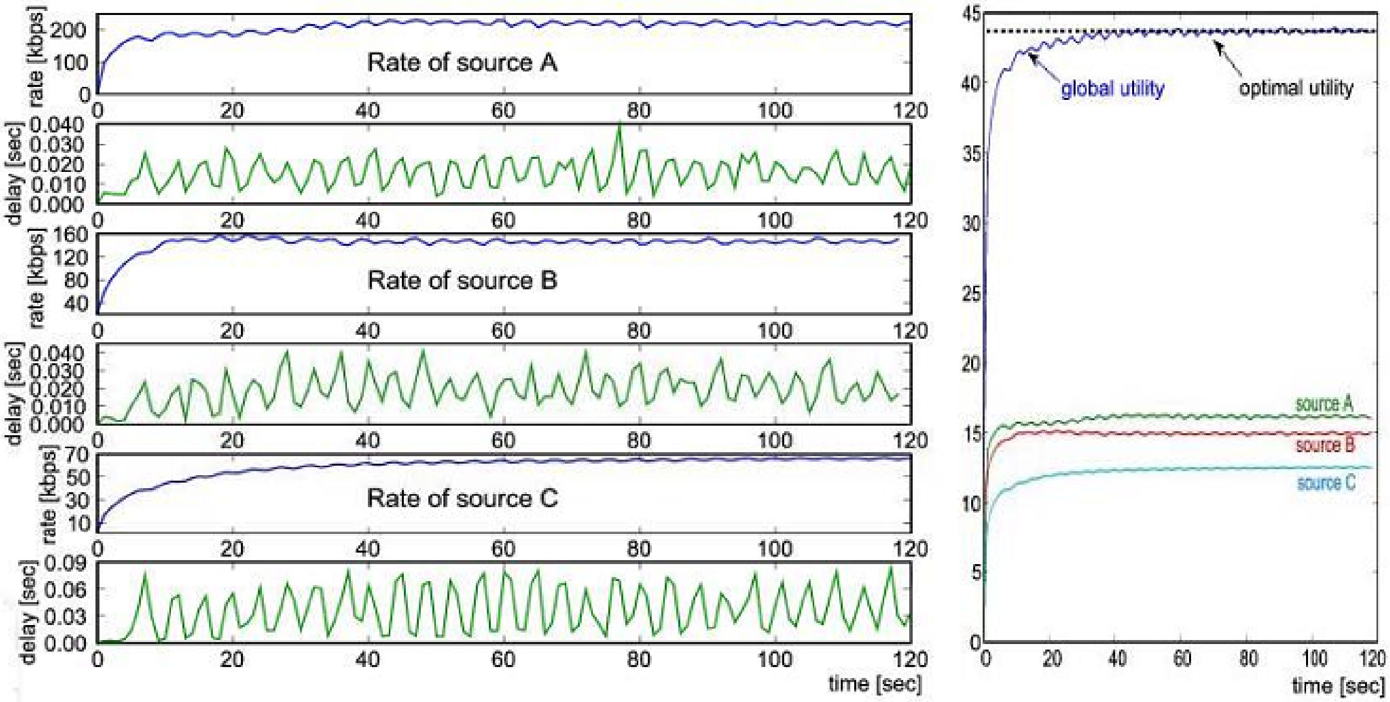
Compute relative OWD between A and B

Compute relative OWD between B and C

An OWD report at most hops one extra peer (helper case)

Internet experiments

- Three peers across US continental: Bay area, Illinois, NYC
  - Uplink capacities: 384, 256, 128 Kbps
  - Estimated one way delay: 40, 20, 33 ms
  - Average packet delivery delay: 95, 105, 128 ms

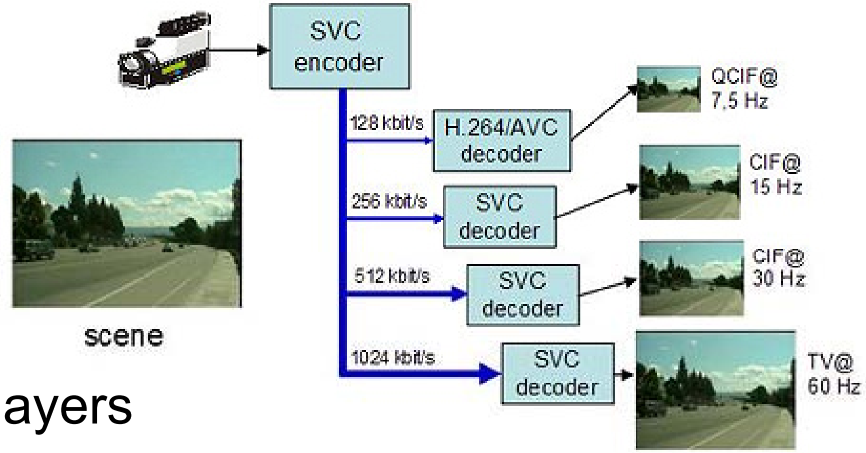


Remarks

- Framework and solution for utility maximization in P2P systems
  - Packing linear number of trees per source is optimal in P2P topology
  - Tree-rate based formulation results in linear constraints
- Distributed algorithms for determining source rates and tree splitting
  - Packet marking based primal algorithm
  - Queueing delay based primal-dual algorithm
- Practical implementation of primal-dual algorithm and Internet experiments

Multi-rate Receivers: Video Coding Model

- Address high variability across peers in
  - Demand for video quality
  - Resources contributed to the system (e.g., uplink)
- Two common approaches
  - Multiple Description Coding (MDC)
  - Layered Coding
- Use layered coding here
  - Scalable Video Coding
  - Base video layer and progressive enhancement layers
  - Necessary to receive all previous layers for additional enhancement layer to be useful



SVC (e.g., H.264/AVC Annex G)

Layered Coding

- x^s_r : receiver r's receiving rate for source s' video
- R_s : set of receivers for source s
- Suppose $x^s_{i_1} \le x^s_{i_2} \le \ldots \le x^s_{i_{|R_s|}}$
- Construct $|R_s|$ multicast sessions
  - Base layer (layer 0) has rate $x^s_{i_1}$ multicasted from s to all receivers in R_s
  - Enhancement layer ℓ has rate $(x^s_{i_{\ell+1}} - x^s_{i_\ell})$ multicasted from s to all receivers in $\{i_{\ell+1}, i_{\ell+2}, \ldots, i_{|R_s|}\}$ $(1 \le \ell \le |R_s| -1)$
- G^s_ℓ : set of receivers for layer l of source s
  - Determined by ordering of the x^s_r values
  - Will be denoted by $G^s_\ell(\{x^s_r\})$

Questions to address

- What is the achievable rate region for receiver rates $\{x^s_r\}$ subject to node uplink constraints?
  - Network coding can be used to mix packets belonging to the same layer of same source only
- How to find a point (choice of rates) in this rate region that is optimal with respect to receiver utilities?

Rate Region B with Intra-session Coding

Traffic on link e due to routing of layer l of source s

$$\sum_{e \in E^+(i)} y_e^{s\ell r} - \sum_{e \in E^-(i)} y_e^{s\ell r} = \begin{cases} +z_\ell^s & \text{if } i = s \\ -z_\ell^s & \text{if } i = r \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

(flow balance constraints)

$$\forall\, i \in N, r \in G_\ell^s(\{x_r^s\}), 0 \leq \ell \leq |R_s| - 1, s \in S$$

$$\sum_{s \in S} \sum_{\ell=0}^{|R_s|-1} \max_{r \in R_s}(y_e^{s\ell r}) \leq C_e \;\; \forall\, e \in E \quad (2)$$

(uplink capacity constraints)

$$z_0^s = \min_{r \in R_s}(x_r^s) \;\; \forall\, s \in S$$

$$z_\ell^s = \overset{\ell+1}{\min_{r \in R_s}}(x_r^s) - \overset{\ell}{\min_{r \in R_s}}(x_r^s)$$

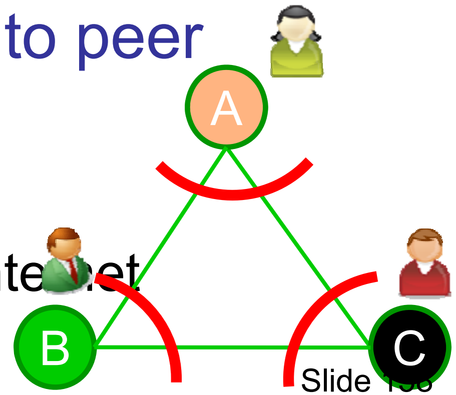
$$\forall\, 1 \leq \ell \leq |R_s| - 1, s \in S$$

Max term models intra-layer network coding

Rate assigned to layer l of source s

Problem Formulation

- Source s transmitting at rate x^s_r to receiver $r \in R_s$
- $U^s_r(x^s_r)$: (concave) utility of receiver r associated with video stream of source s
  - Depends on receiver's window size/screen resolution
  - Depends on amount of delta change across frames in video of source s
  - Example: PSNR curve
- Only uplinks of peers are bottleneck links
- Maximize total utility of all receivers subject to peer uplink constraints
  - Joint rate allocation and routing problem
  - Need distributed solution for deployment in the Internet



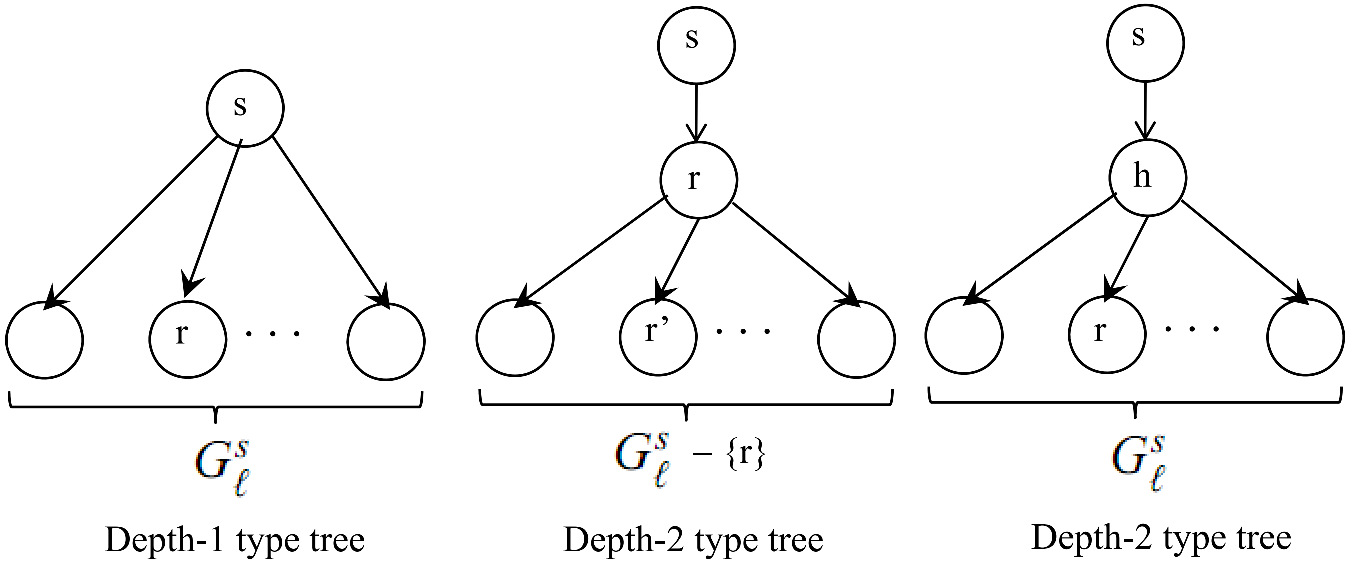
Multi-rate Multicast Utility Maximization

$$\max_{\boldsymbol{x}} \quad \sum_{s \in S} \sum_{r \in R_s} U_r^s(x_r^s)$$
$$s.t. \quad \boldsymbol{x} \in \mathcal{B}.$$

- S: set of sources
- R_s : set of receivers for source s
- B is the feasible region for rates $\{x^s_r\}$
  - Only peer uplink capacities are bottleneck
  - Allow intra-layer network coding

Rate Region for Multi-source Multi-rate Multicast with Layered Coding

- Node uplink capacities only, multi-source, layered coding
  - Routing along linear number of trees for each layer achieves rate region B



G_ℓ^s : set of receivers for layer l of source s

How is rate region B achieved?

- High-level idea: Decompose B into sub-regions with a given ordering of receiver rates per source
  - Suppose we know the ordering of receiver rates x^s_r , $r \in R_s$ for each source s, denoted by $\pi = (\pi^s , s \in S)$
  - $B(\pi)$: subset of rate region B where receiver rates are ordered according to π
  - Observe that $B = U_\pi B(\pi)$
- Theorem 1: *The rate region $B(\pi)$ can be achieved by packing depth-1 type and depth-2 type trees.*
  - Number of trees per source (= $|N||R_s| - \frac{|R_s|(|R_s|-1)}{2} - 1$) is at most quadratic in total number of peer nodes
- Theorem 2: *The optimal solution in rate region* B *can be expressed as a linear superposition of flows along depth-1 type and depth-2 type trees for every source s.*

Receiver-independent utility functions

- Theorem 3: *If $U^s_r = U^s$ for all $r \in R_s$, $s \in S$, then there exists an optimal solution in which $x^s_r = x^s$ for all $r \in R_s$, $s \in S$ (receiver rates are identical for same source).*

Tree-based Multi-rate Multicast Utility Maximization

$$\max_{\xi} \quad \sum_{s \in S} \sum_{r \in R_s} U_r^s \left(\sum_{m: m \in s, r \in m} \xi_m \right)$$

$$s.t. \quad \lambda_e \le C_e, \quad \forall e \in E.$$

- ξ_m : rate on tree m
- λ_e : aggregate rate on uplink e

$$\lambda_e = \sum_{s \in S} \sum_{m: m \in s, e \in m} b_e^m \xi_m, \quad \forall e \in E$$

- b^m_e : number of branches of tree m that pass through uplink e
- (Simpler) Tree-rate based formulation which is amenable for solution using distributed rate control algorithms

Ordering of Receiver Rates

- Tree-rate based formulation assumes that ordering of receiver rates for every source is known
- How can an ordering be obtained in practice?
  - In order of receiver uplink capacities: peers who contribute more to the system receive better quality video
  - In order of receiver utility coefficients
  - Peer individual preference: The stream being currently focused on by the receiver should be of higher resolution than the other streams
  - Human communication dynamics: If peer A is talking to peer B with eye-gaze, then source A video should be sent at high resolution to receiver B

Queueing Delay Based Primal-Dual Algorithm

- Lagrangian multipliers p_e for each uplink e

Lagrangian multipliers

$$L(\xi, p) = \sum_{s \in S} \sum_{r \in R_s} U_r^s \left(\sum_{m: m \in s, r \in m} \xi_m \right) - \sum_{e \in E} p_e (\lambda_e - C_e)$$

- Primal-dual algorithm

$$\dot{\xi}_m = k_m \Bigg(\overbrace{\sum_{r \in m} U'^s_r \left(\sum_{m: m \in s, r \in m} \xi_m \right)}^{\text{Incentive to increase rate on tree m}} - \underbrace{\sum_{e \in m} b_e^m p_e}_{\text{Aggregate queueing delay on tree m}} \Bigg)^+_{\xi_m}$$

$$\dot{p}_e = \frac{1}{C_e} (\lambda_e - C_e)^+_{p_e},$$

- p_e can be interpreted as queueing delay on peer uplink e
  - Provided as feedback to every source from all of its receivers

$(a)_b^+ = a$ if $b > 0$, and is $\max(0, a)$ otherwise

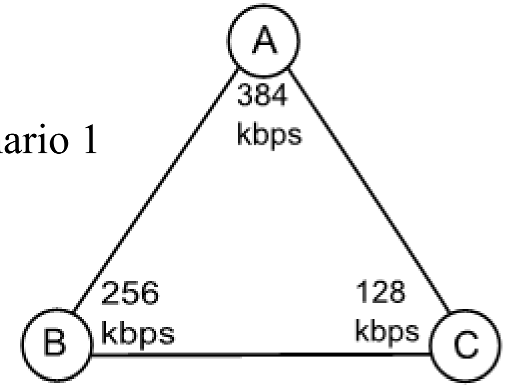
Distributed Properties of Rate Control Algorithms

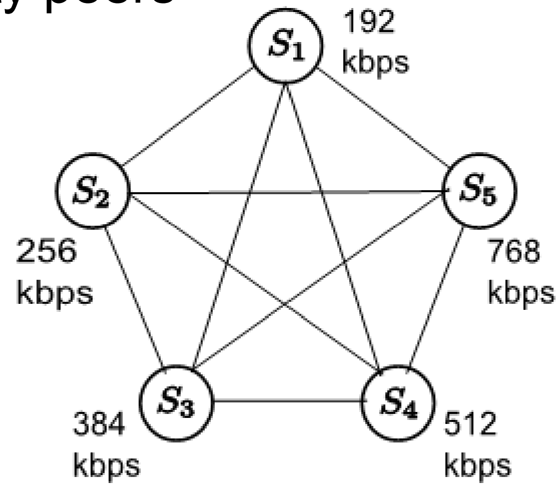
- Does not require global knowledge of
  - Network conditions
  - Peer uplink capacities
  - Utility functions of other sources' receivers
- Adapts to uplink cross-traffic
- Adapt to changes in utility function (user moving or still)

Experimental Evaluation

- Peers running on virtual machines in a lab testbed
- Uplink capacity emulation through rate limiting
- Queueing delay based primal-dual algorithm
- Two peer scenarios
  - Scenario 1: 3 peers, receiver-independent utility functions
  - Scenario 2: 5 peers, diverse utility peers

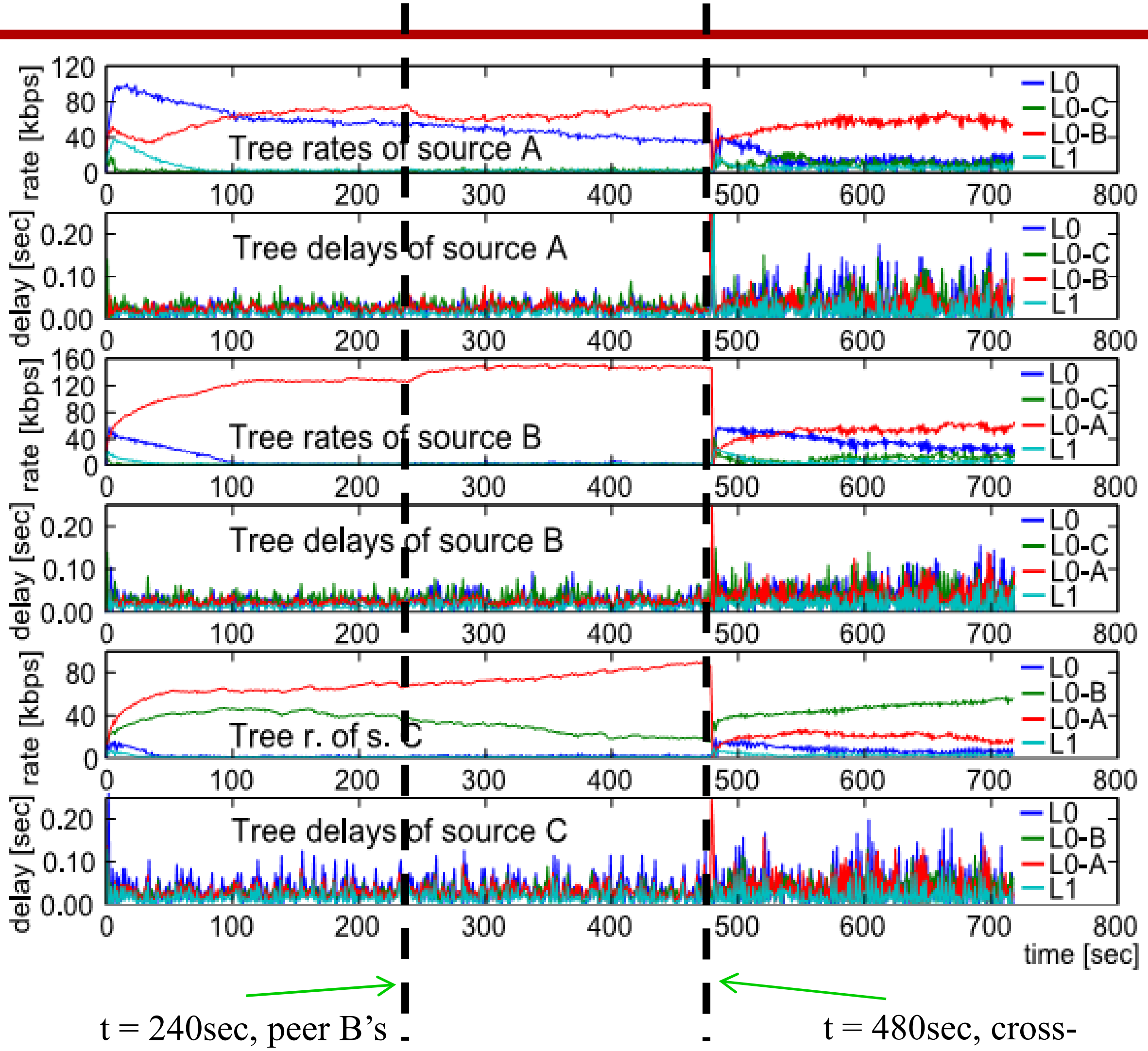
Scenario 1





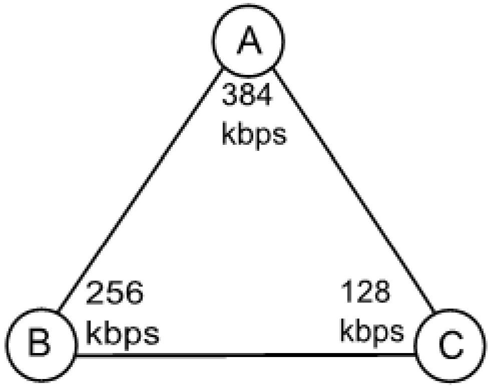
Scenario 2

Tree rates in Scenario 1

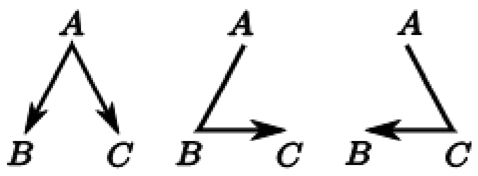


t = 240sec, peer B's utility coefficient increases

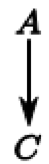
t = 480sec, cross-traffic initiated at peer A



Layer 0 trees

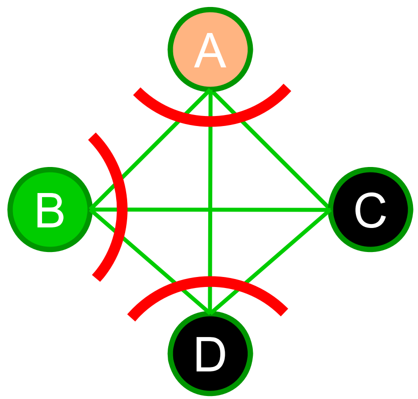


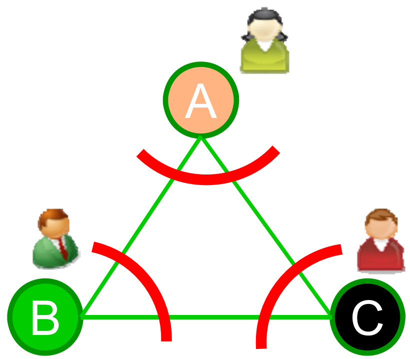
Layer 1 tree



Summary and Takeways

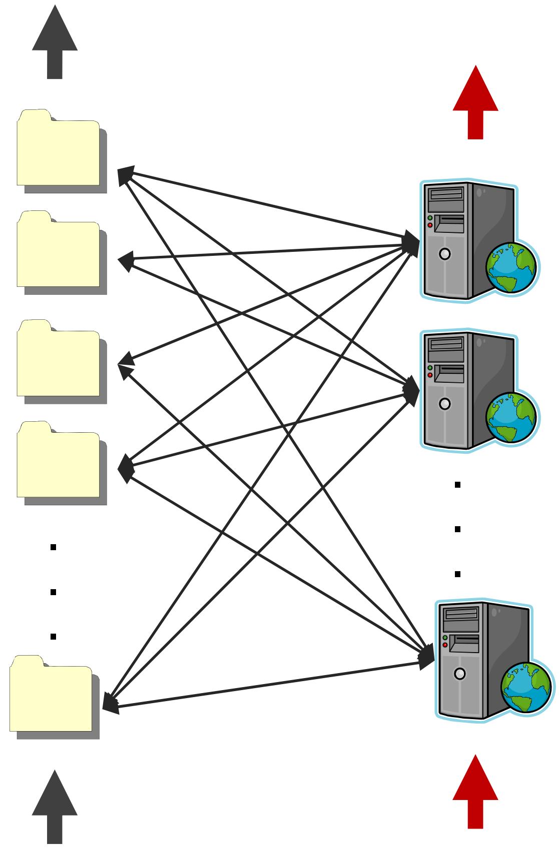
- Utility maximization based approach for multi-source multi-rate peer-to-peer communication scenarios
- Layered coding based video distribution
- Sufficient to use at most quadratic number of trees per source to achieve rate region
- Distributed algorithms for tree-rate control





IV. QoS in Dynamic Peer-to-Peer Systems

QoS Is Important for P2P Systems



Users dynamically arrive
each fetches a file and leaves

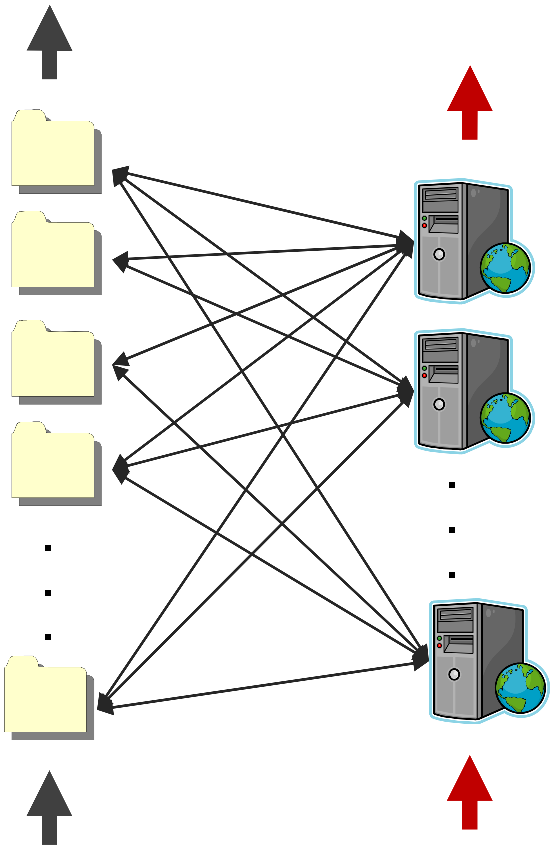
Servers dynamically arrive
each serves for a while and leaves

Example: A P2P storage system

- Users store private files on peer PCs and download them later

- Advantages:
    - High throughput (download from neighbors)
    - ISP also benefits (sell the reach-ability of peer PCs)
    - Cost effective (to-be-invest.)

QoS Is Important for P2P Systems



Queuing analysis helps to answer

- Is the user waiting time finite?
- What is average user waiting time?
- What is the impact of server dynamics?
    - Different level of dynamics maps to different storage systems

A 3-pages detour on classical queuing models

Users dynamically arrive
each fetches a file and leaves

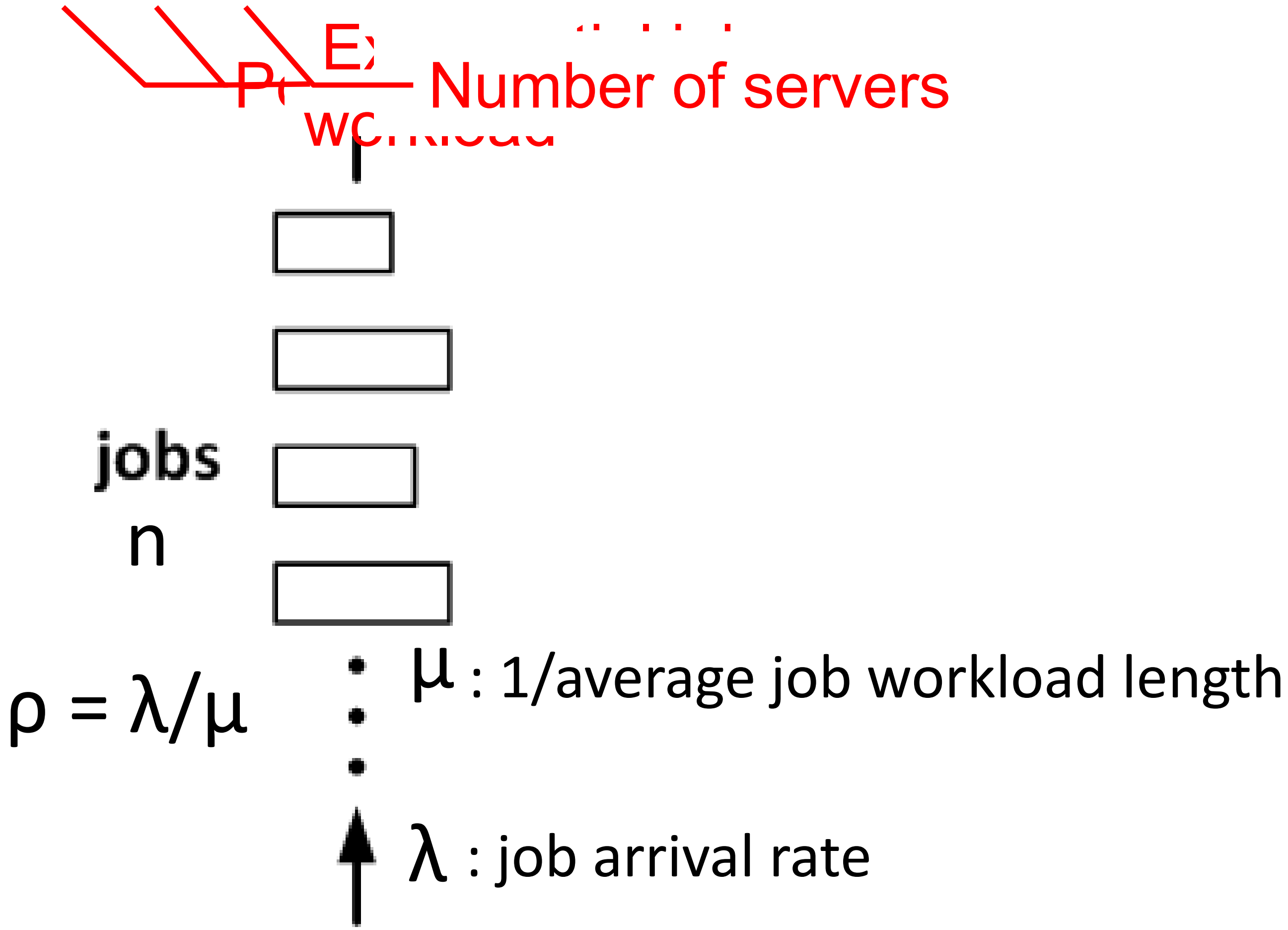
Servers dynamically arrive
each serves for a while and leaves

A Brief History of Queuing Theory

❖ Problem formulation	A. K. Erlang, 1909
❖ Loss rate and waiting time	A. K. Erlang, 1917
❖ Notation A/B/s	D. G. Kendall, 1953
❖ Little's Law	J. D. C. Little 1960
❖ Round robin, process sharing	L. Kleinrock, 1960s
▪ Application to computer systems	
❖ Application to computer networks	1980-90's
❖ Application to P2P systems	2000's
❖ …	

An Example of Classical Queuing Model

- M/M/s model



An Example of Classical Queuing Model

- Stability
  - If ρ < s, then all arriving jobs will be cleared in finite time
  - Positive Recurrence of Markov Chain

- Average job waiting time (Little's Law)

$$D_{M/M/s} = \frac{1}{\mu}\frac{1}{s-\rho}$$

- Similar results can be obtained for M/G/s and G/G/s models

Unique Features of P2P Service Systems

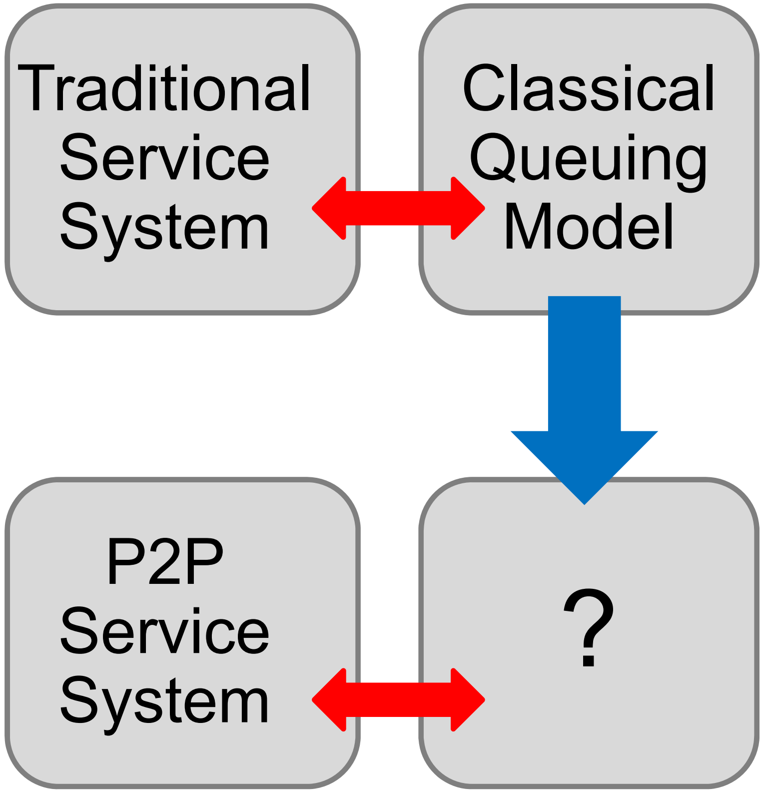
Classical service system

- Dynamic arriving jobs
- (Mostly) Static servers
- Limited study on dynamic servers
  - Server vacation/repair models

P2P service system

- Dynamic arriving jobs
- Dynamic arriving servers
- Server dynamics and job-server correlation is the new ingredient

Focus



- General P2P queuing model
  - A taxonomy and notations
- Answer old question
  - Stability Condition (finite waiting time)
- Exploring new territories
  - Impact of server dynamic
  - Impact of job-server correlation

Extended Queuing Model

- M/M/s --> M/M/(M/M)

Poisson job arrival

Exponential job workload

Fixed (s) servers

Poisson job arrival

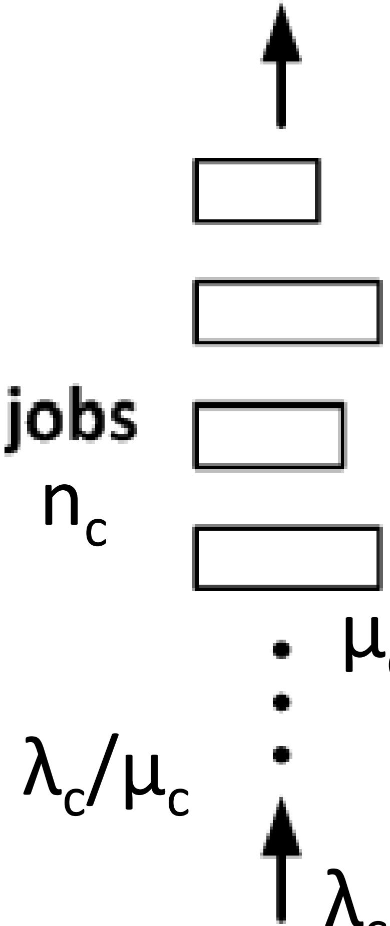
Exponential job workload

Poisson server arrival,

Exponential server lifetime

- Model server dynamics in p2p systems
- Introduce new ingredients
  - Different server dynamic
  - Job-server correlation

P2P Storage: M/M/(M/M) Queue



μ_s :
1/server life time

n_s

μ_c : 1/average job workload length

$\rho_c = \lambda_c/\mu_c$

$\rho_s = \lambda_s/\mu_s$

λ_c : job arrival rate

λ_s :
server arrival rate

- Job arrival and server dynamics are independent

Stability of M/M/(M/M) Queue

- A M/M/(M/M) queuing system is stable if and only if $\rho_c < \rho_s$.
  - Model as a 2-D Markov Chain (not time rev.)
  - ρ_c: job workload
  - ρ_s: the service capacity
- M/M/s as a special case: $\rho_c < s$

- Proof idea:
  - Model as a Quasi Birth Death process. Apply matrix analytical method (by M.F. Neuts, 1970s)
  - Alternative: construct a foster Lyapunov function

Verification via Simulation

- M/M/(M/M) queue
- Fix λ_s=0.005, μ_s=0.0005, and μ_c=0.0006
- A

Stability of M/G/(M/M) Queues

- Stability condition for M/G/(M/M) is also $\rho_c < \rho_s$
  - Prove by constructing a Foster-Lyapunov function
  - Workload distribution from real file size distribution

Job-Server Correlation

- Job dynamics may correlate with server dynamics

Job-server dynamics negative correlated

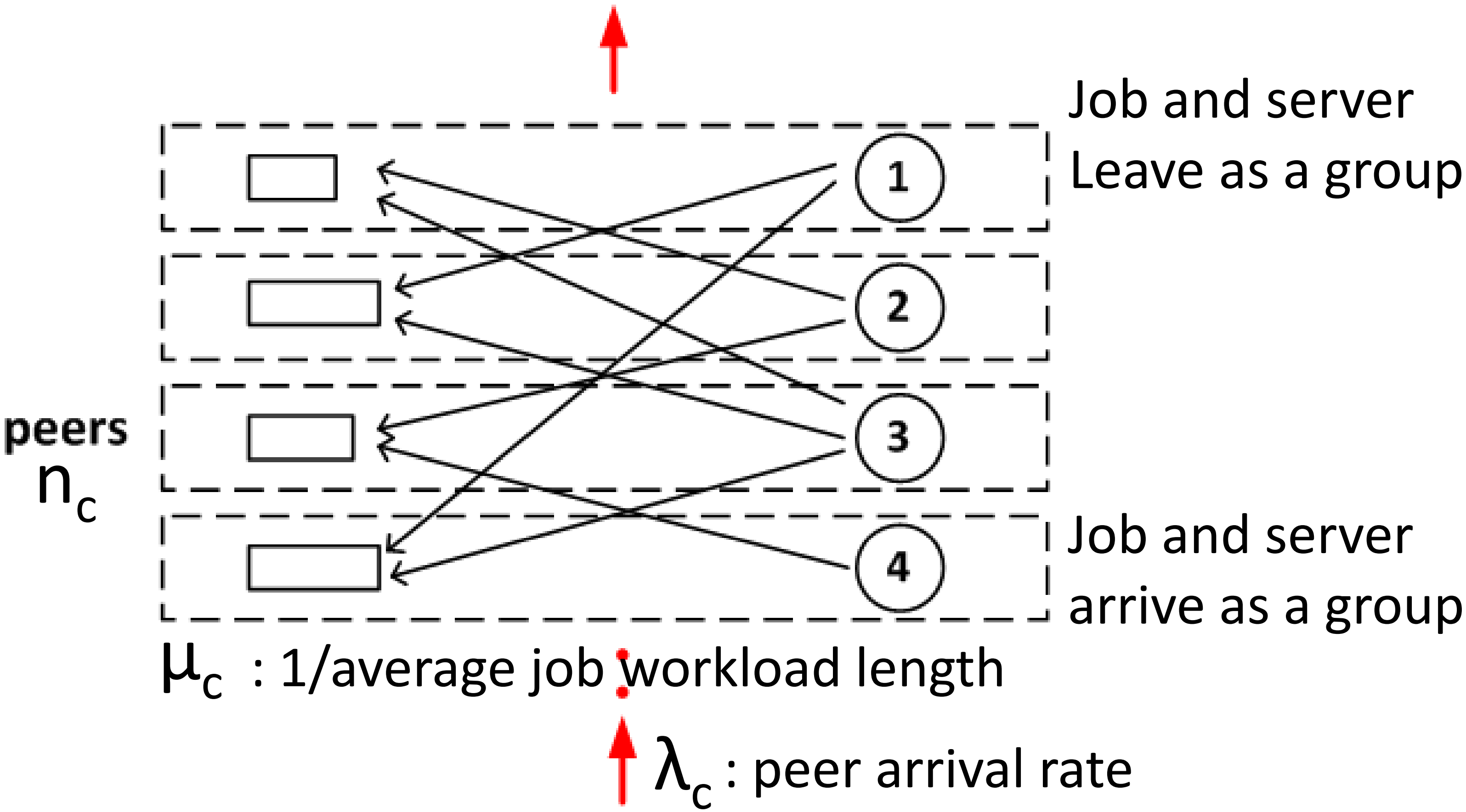
M/M/(M/M)
Independent

M/M/(-/-)
Identical job-server dynamics

Complete spectrum

P2P Download: Identical Job-Server Dynamics

- M/M/(-/-) queue (example: P2P file sharing)



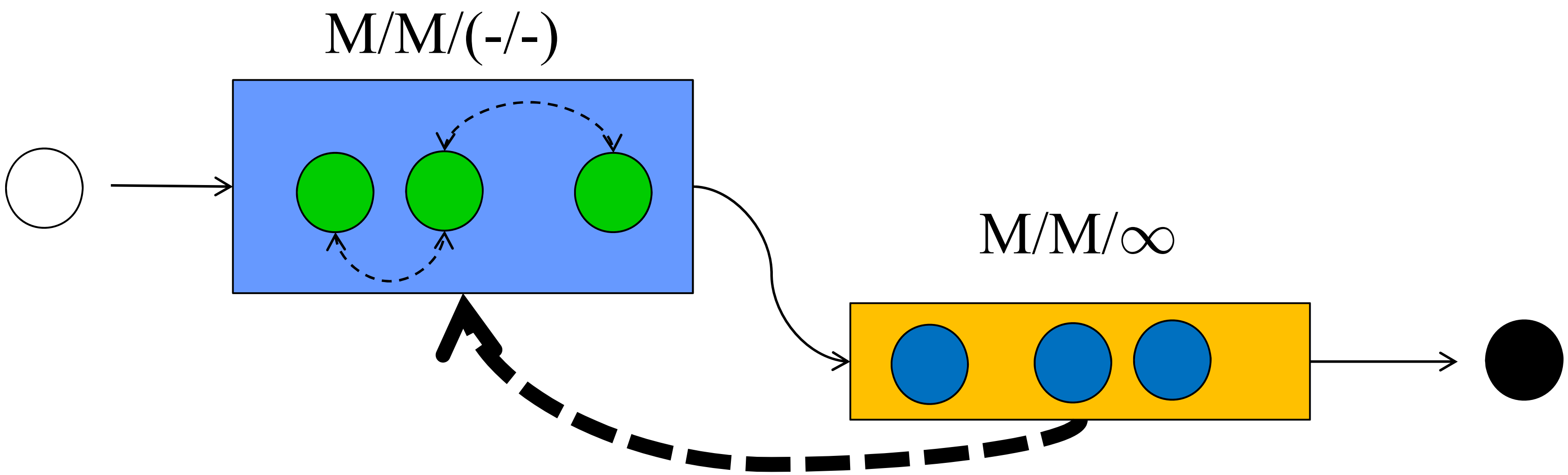
Stability of M/M/(-/-) Queue

- M/M/(-/-) queue is always stable
  - A job brings in a finite workload but a service capacity increasing linearly in time
  - In finite time we have capacity exceed workload

- Proof idea: Reduce to a M/M/∞ queue

Modeling Bit-Torrent Like Systems [Qiu-Srikant 04]

- One M/M/(-/-) queue for downloading peer swarms
- One M/M/∞ queue for seeder swarms



- Assumes one class of peers; study equilibrium performance (stability and delay)

Modeling Bit-Torrent Like Systems [Fan-Chiu-Lui 06]

- Extend [Qiu-Srikant 04] to multiple classes of peers, and study download-time-fairness trade-off

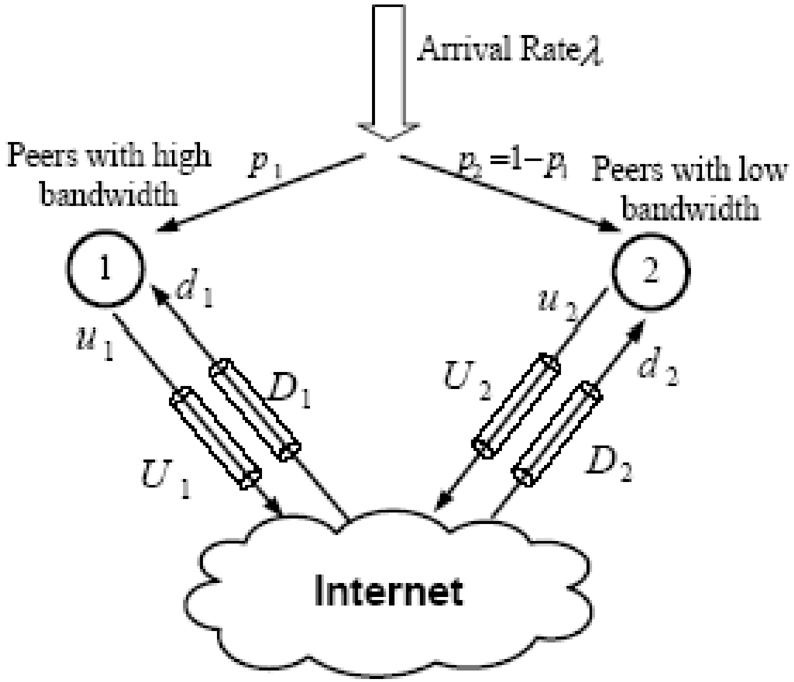
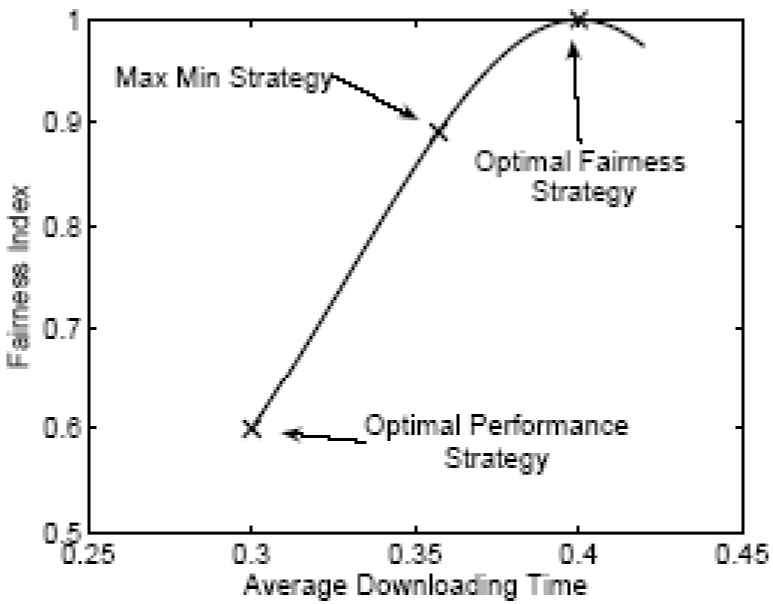
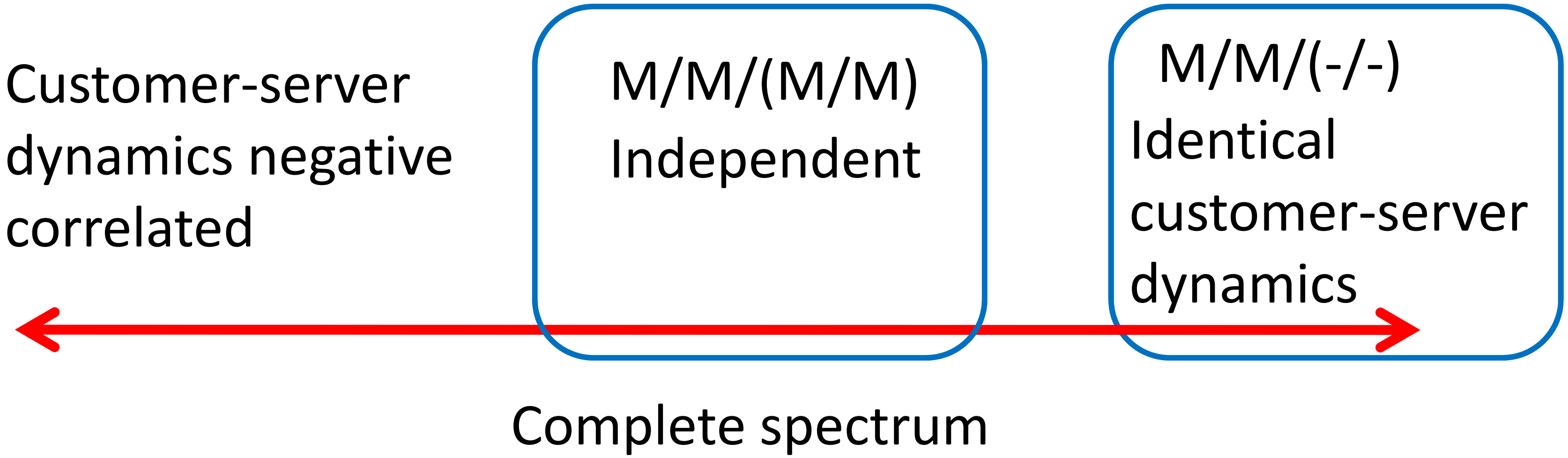


Fig. 1. The System Model



Job-Server Correlation

- Job dynamics may correlate with server dynamics



Stability condition under general correlation: open

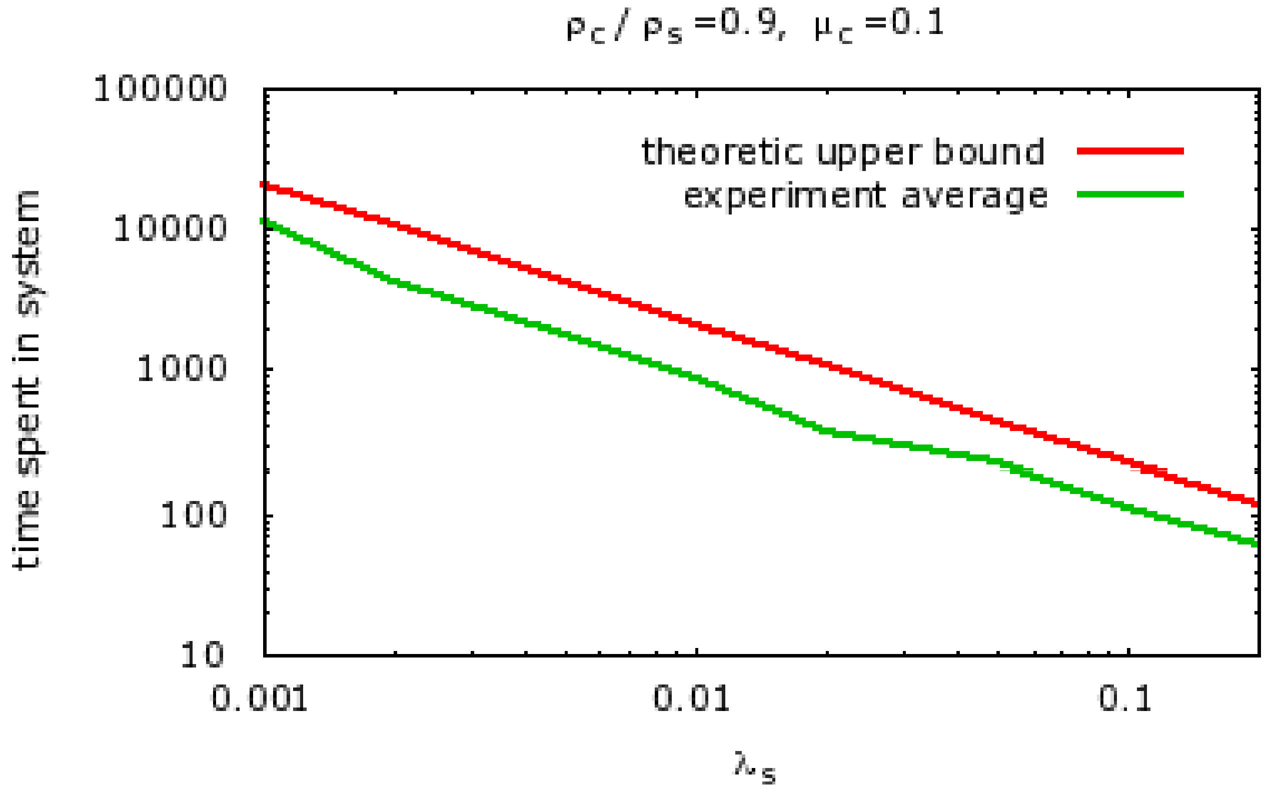
Average Waiting Time Analysis

- Stability is not enough: only say waiting time is finite

- Average waiting time
  - Little's law $D = \frac{1}{\lambda_c} E[n_c]$
  - Challenging to find $E[n_c]$ due to the Markov Chain is not time reversible
  - Still an open problem
  - Study via simulations

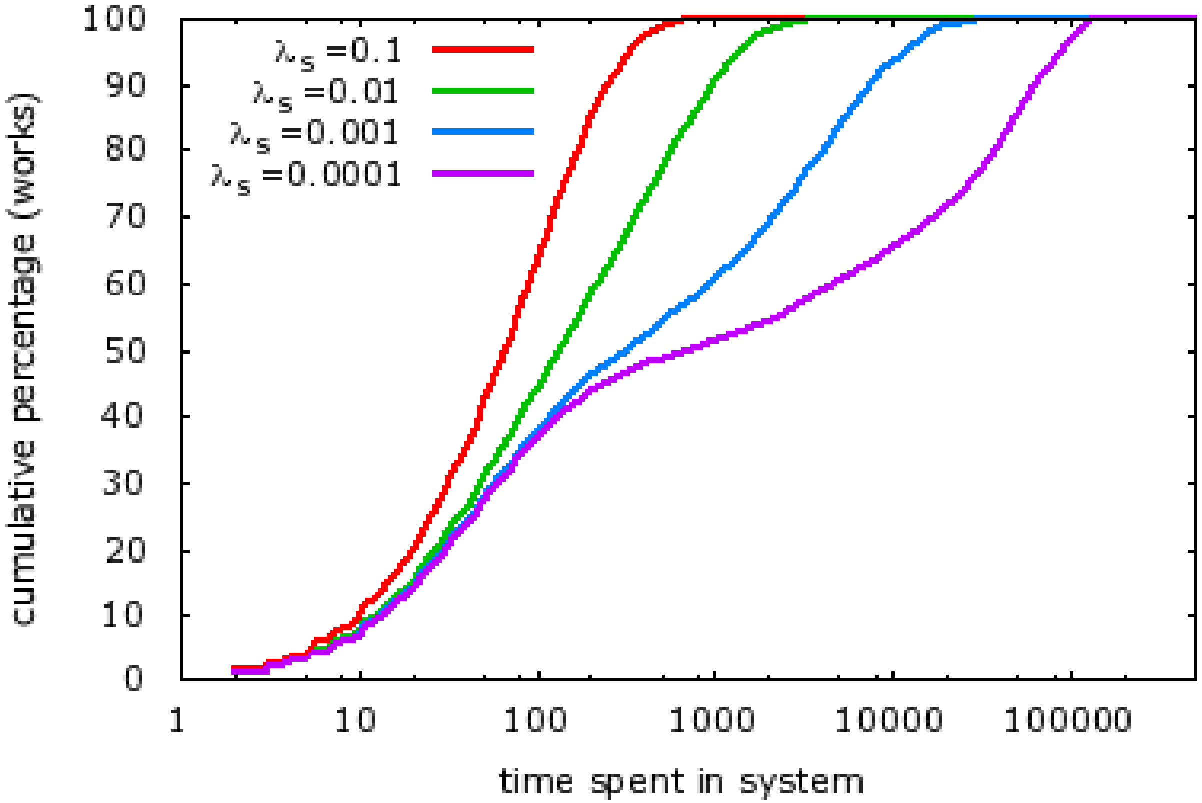
Impact of Server Dynamics: Simulation

- M/M/(M/M)
- Fix ρ_c (fix λ_c and μ_c)
- Fix ρ_s (vary λ_s and μ_s proportionally)



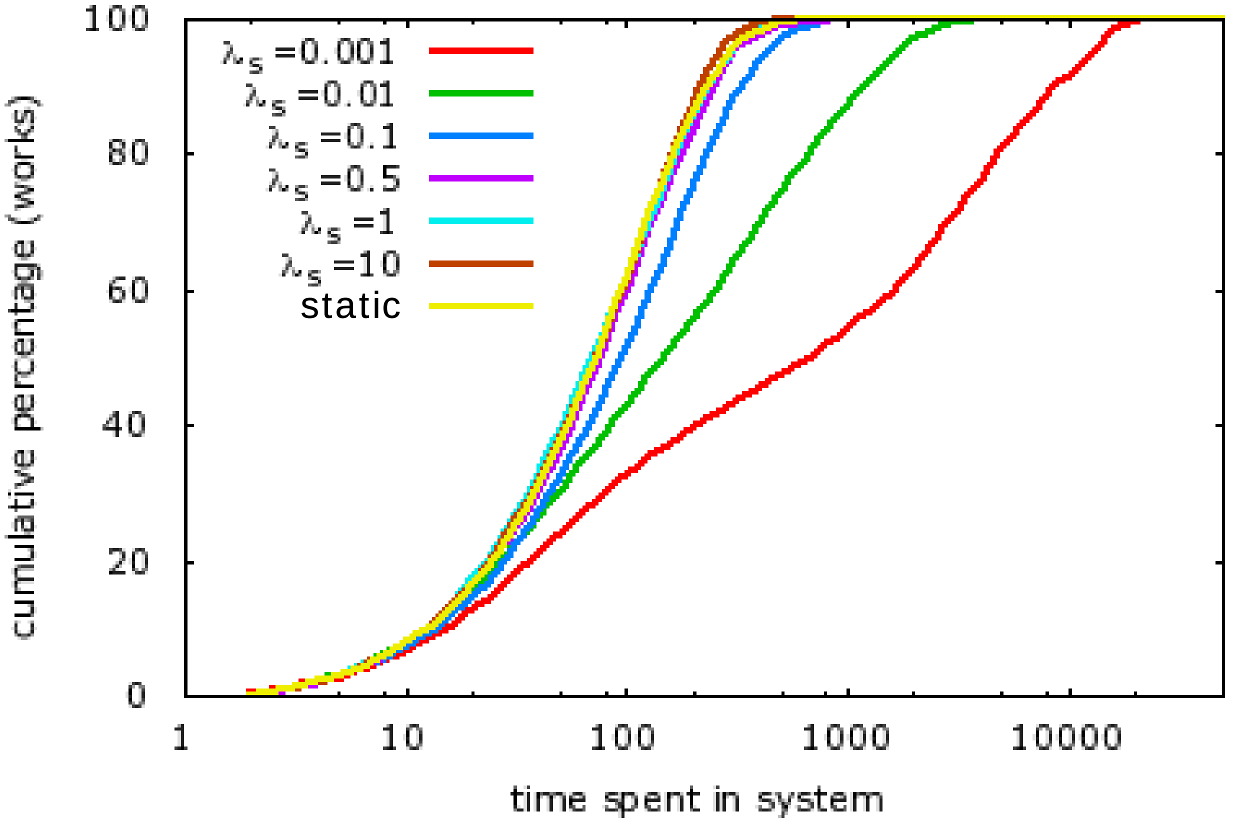
Impact of Server Dynamics: Result

❖ Higher server dynamics leads to shorter waiting time



Impact of Server Dynamics: Compare with Static System

- Static system is a limit of dynamic system when system dynamic increases

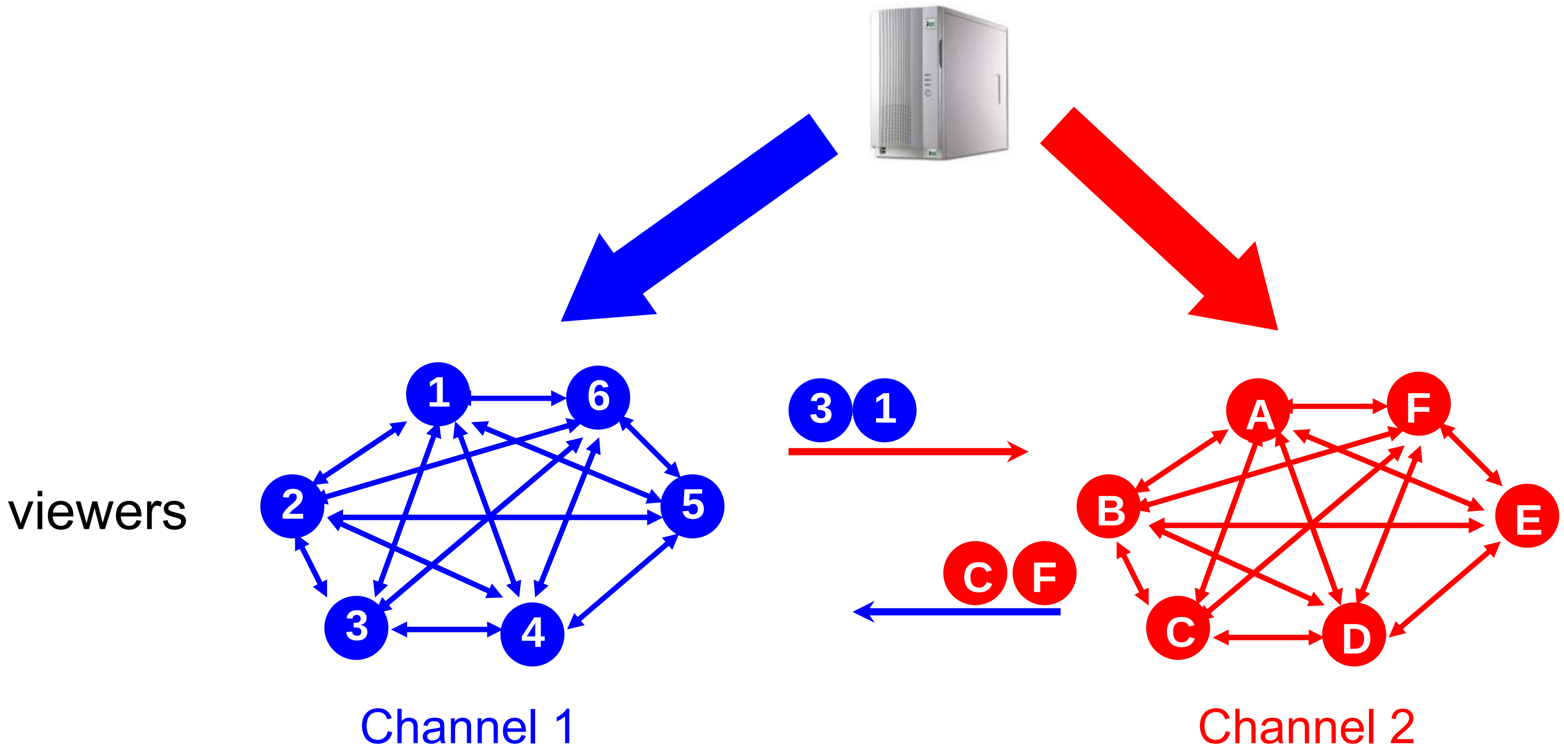


Networks of P2P Queuing Systems

- One P2P system is one P2P Queue
- For example, one channel in P2P streaming system: one P2P queue
- Multi-channels in P2P streaming systems: A network of multiple P2P queues [Wu-Liu-Ross INFOCOM 2009]

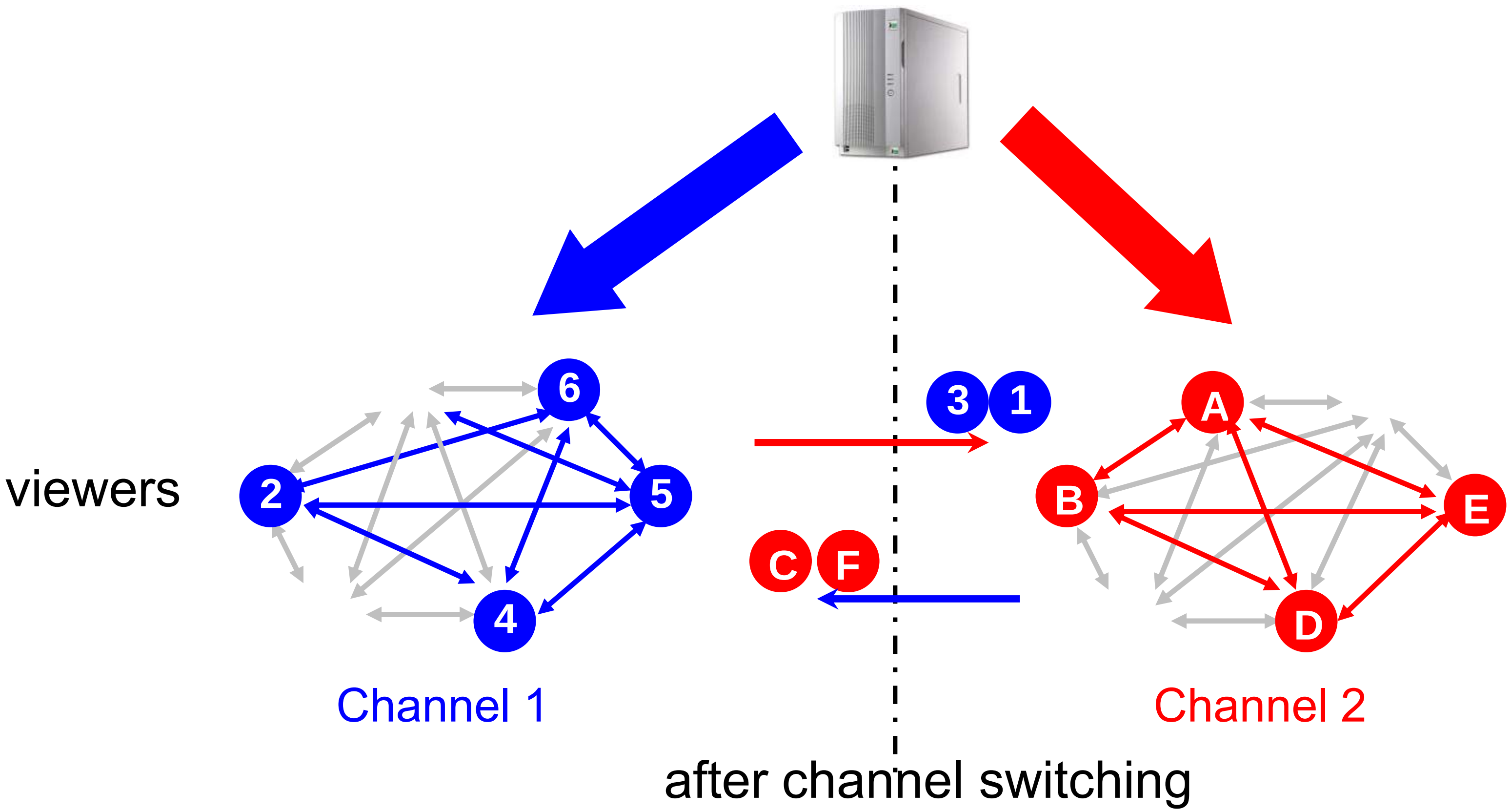
Multi-channel P2P Queuing Networks [Wu-Liu-Ross 09]

- Peers Poisson-ly arrive into one channel, stay for exponential long time, and leave the channel to another channel or depart from the system



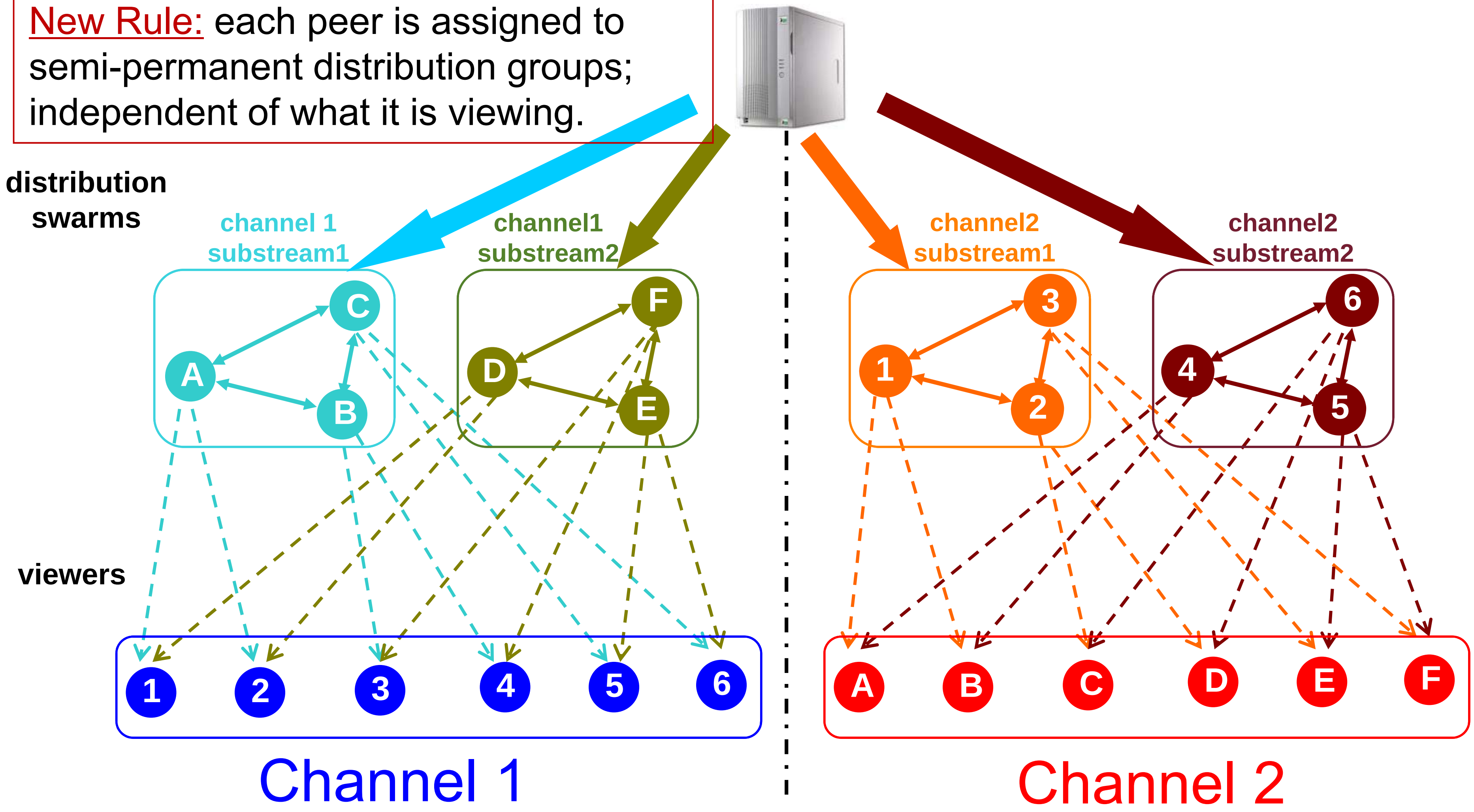
Channel Churn in Isolated Channel Design [Wu-Liu-Ross 09]

Drawback: distribution systems disrupted when peers switch channels



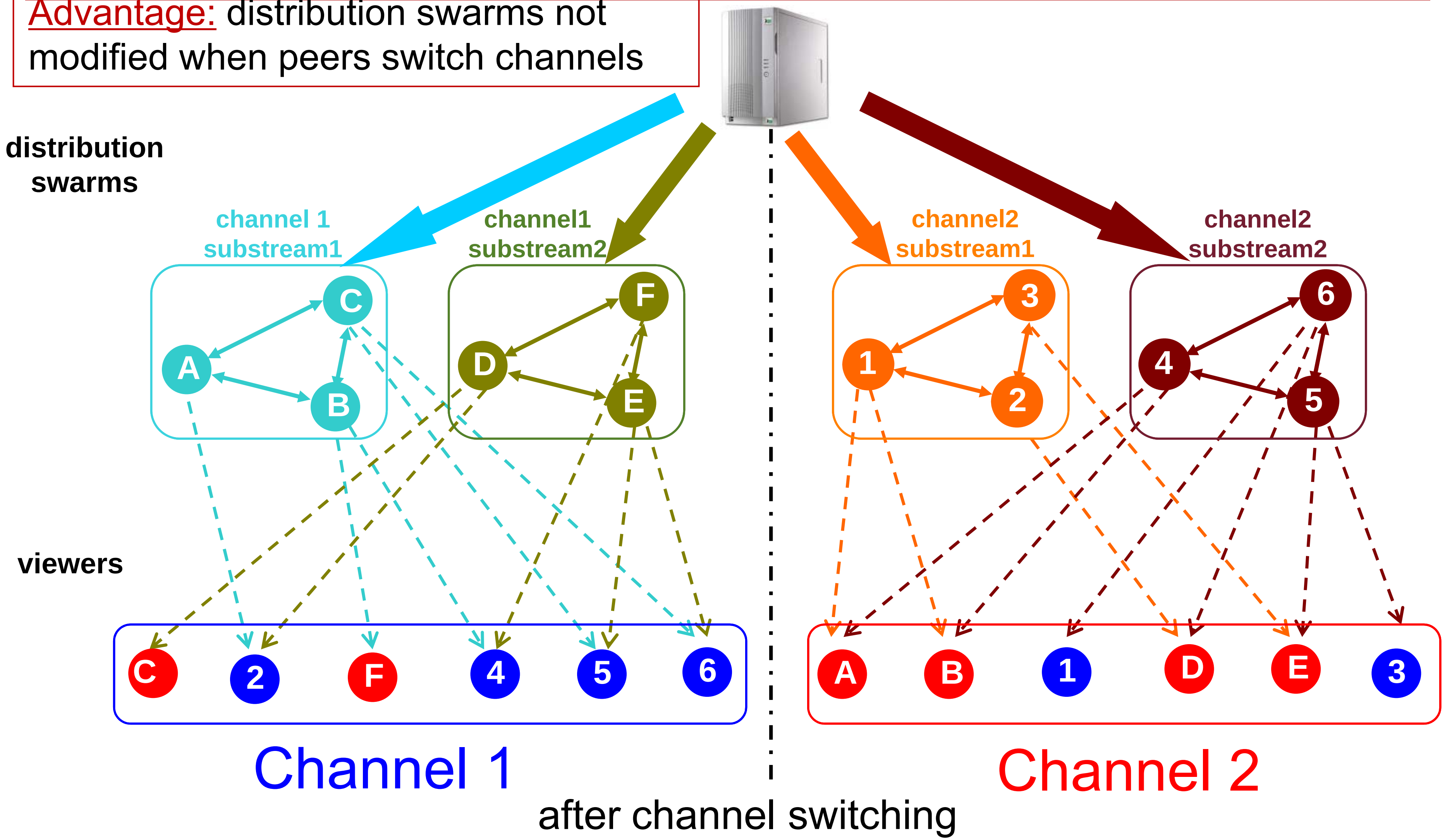
Redesign Multi-Channel System: View-Upload Decoupling [Wu-Liu-Ross 09]

<u>New Rule:</u> each peer is assigned to semi-permanent distribution groups; independent of what it is viewing.

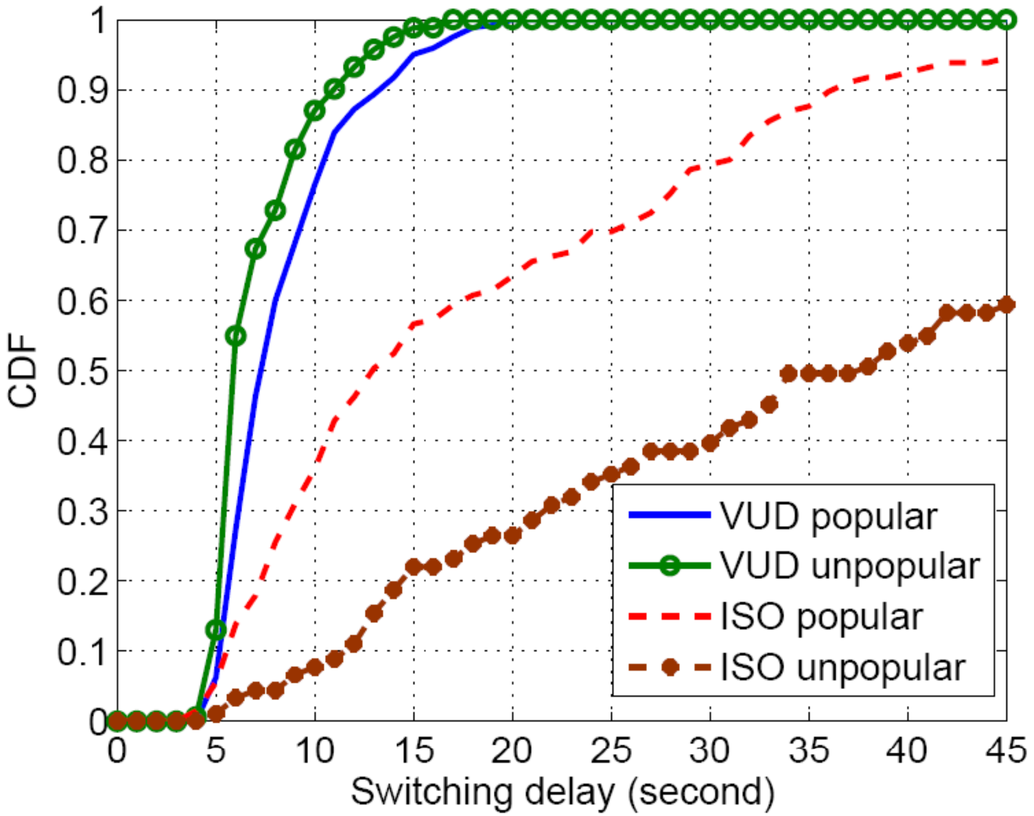


Redesign Multi-Channel System: View-Upload Decoupling [Wu-Liu-Ross 09]

Advantage: distribution swarms not modified when peers switch channels



Performance Gain Shown via Simulation and P2P Queuing Network Analysis

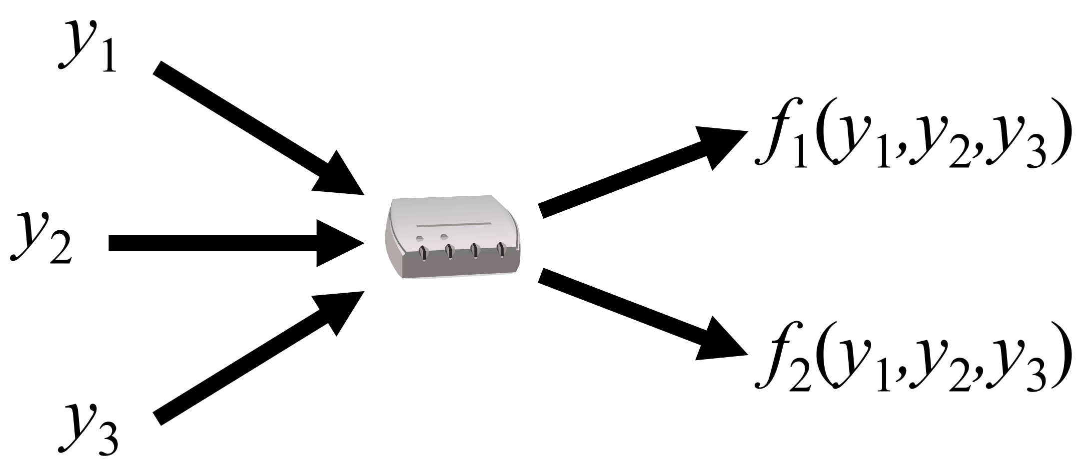


Switching delay = time to acquire 5 seconds of new channel

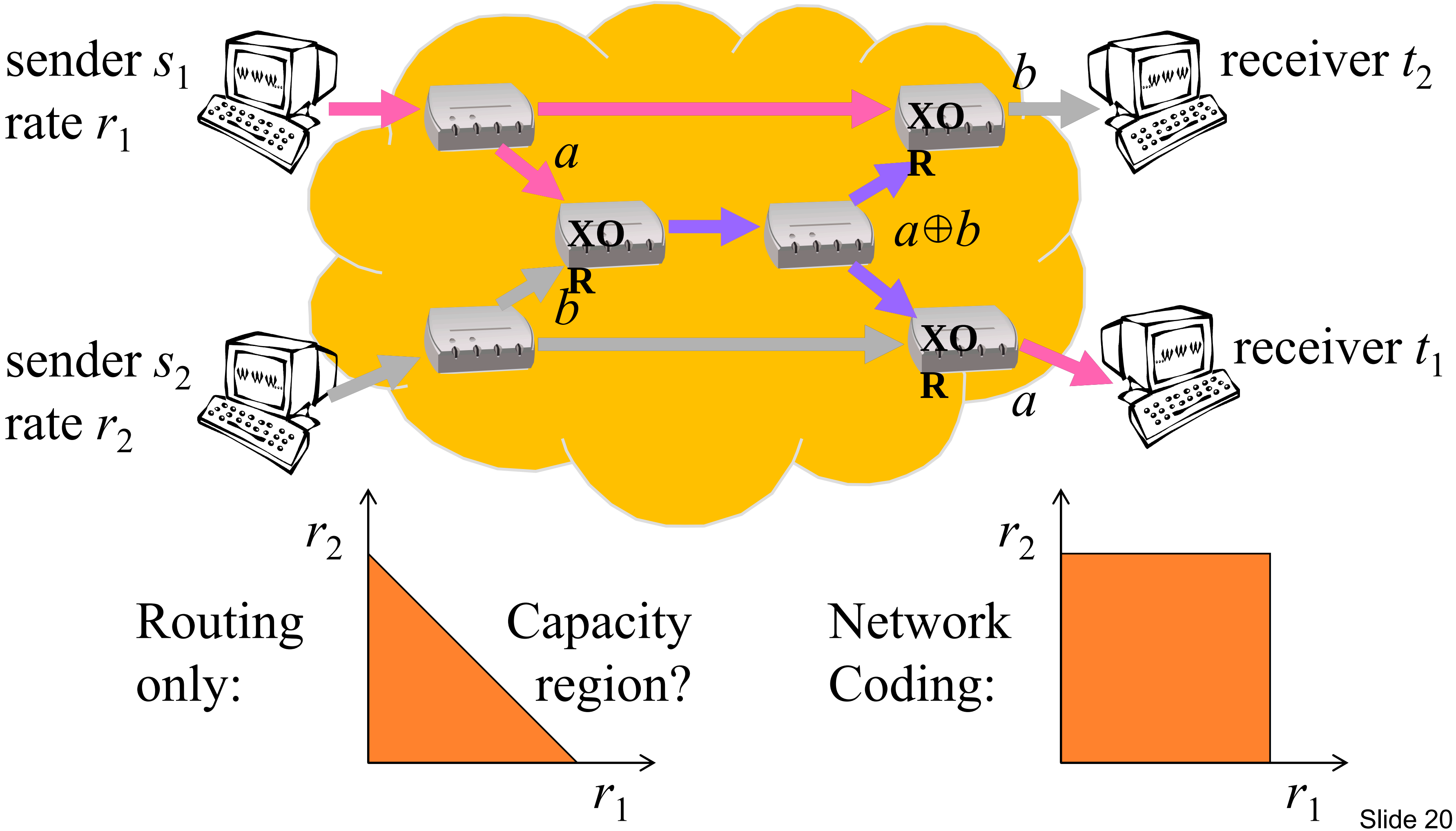
➢ VUD achieves smaller channel switching delay.

V. Network Coding in Peer-to-Peer Systems

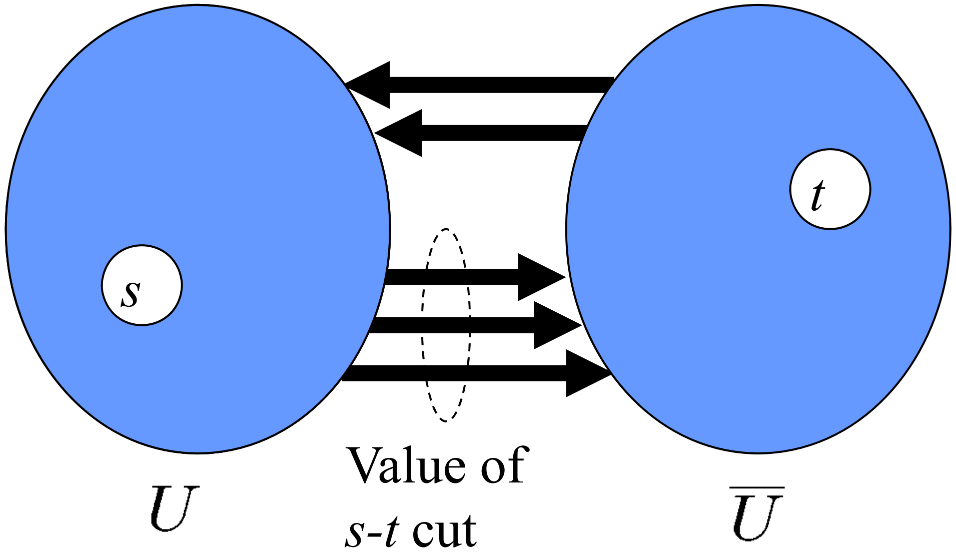
Introduction:
Routing vs Network Coding



Network Coding can Increase Throughput

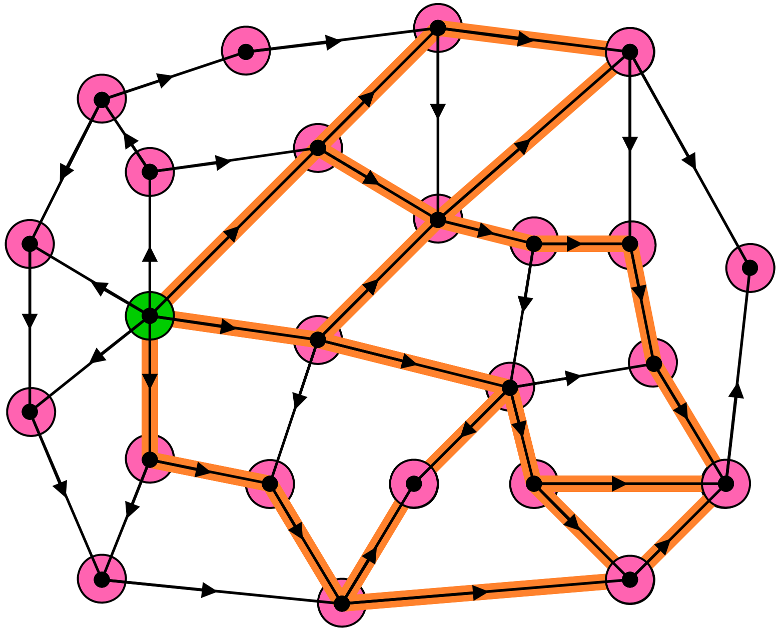


Single Session
– Unicast Case



- rate(s,t) ≤ MinCut(s,t)
- Menger (1927):
  - MinCut(s,t) is achievable, i.e., MaxFlow(s,t) = MinCut(s,t), by packing edge-disjoint directed paths

Single Session – Broadcast Case



Given:

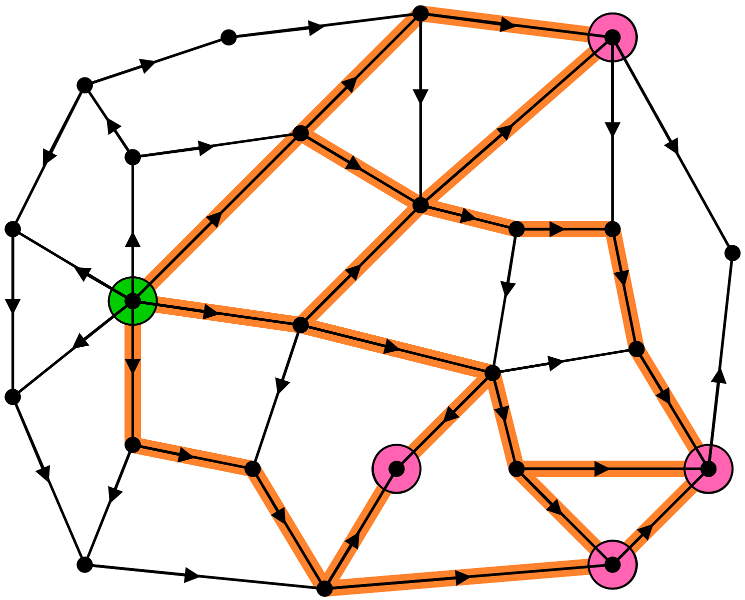
Directed graph (V,E)

Sender s

Receiver set (all other nodes in V)

- rate$(s,V) \le \min_{v \in V}$ MinCut(s,v)
- Edmonds (1972):
  - $\min_{v \in V}$ MinCut(s,v) is achievable ("broadcast capacity") by packing edge-disjoint directed spanning trees

Single Session
– Multicast Case



Given:

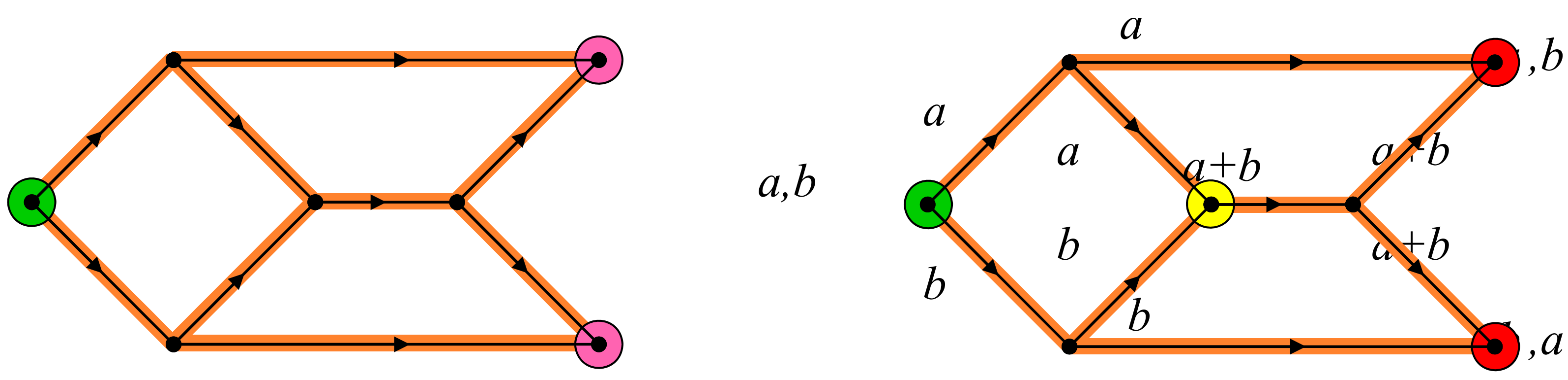
Directed graph (V,E)

Sender s

Receiver set T
(subset of V)

- rate$(s,T) \leq \min_{t \in T}$ MinCut(s,t)
- $\min_{t \in T}$ MinCut(s,t) is NOT always achievable
by packing edge-disjoint Steiner (multicast) trees

Network Coding
Achieves Multicast Capacity



optimal routing
throughput = 1

network coding
throughput = 2

- ❖ Alswede, Cai, Li, Yeung (2000):
  - $\min_{t\in T}$ MinCut(s,t) is always achievable by network coding
  - h = $\min_{t\in T}$ MinCut(s,t) is “multicast capacity”

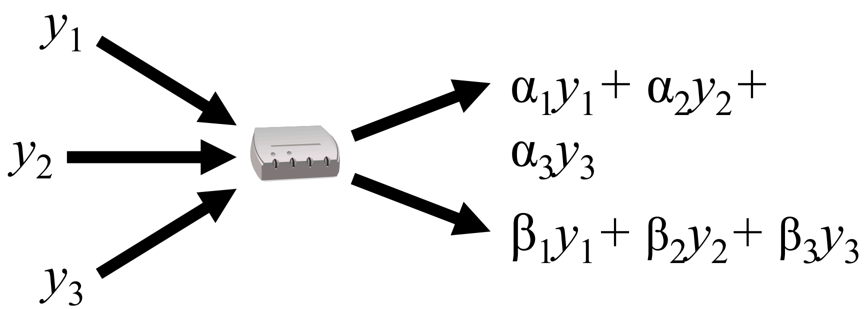
sender

receiver

coding node

Linear Network Coding is Sufficient

- Li, Yeung, Cai (2003) *– IT Best Paper Award 2006*
  Koetter and Médard (2003)
  - Linear network coding is sufficient (to achieve multicast capacity)



- Jaggi, Chou, Jain, Effros; Sanders, et al. (2003)
  Erez, Feder (2005)
  - Polynomial time algorithm for finding coefficients

Making Network Coding Practical

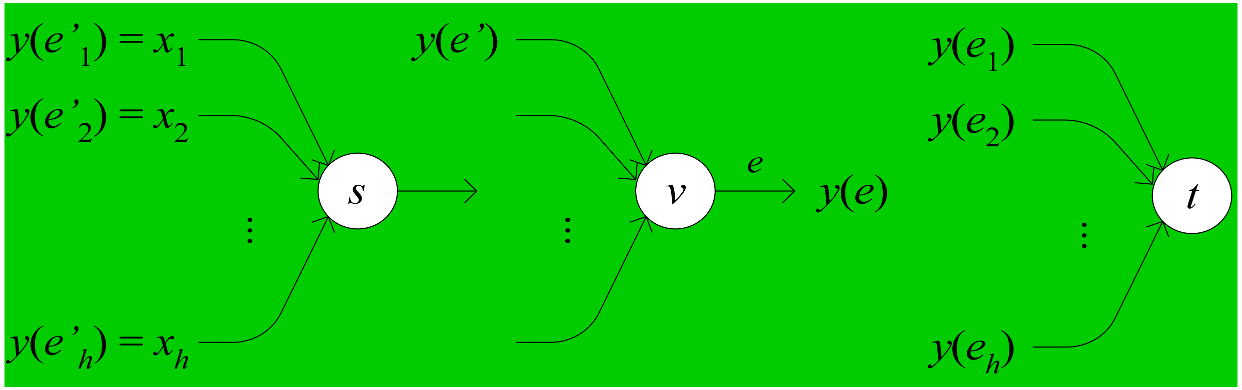
- Packetization
  - Header removes need for centralized knowledge of graph topology and encoding/decoding functions
- Buffering
  - Allows asynchronous packets arrivals & departures with arbitrarily varying rates, delay, loss

[Chou, Wu, and Jain; Allerton 2003]

[Ho, Koetter, Médard, Karger, and Effros, ISIT 2003]

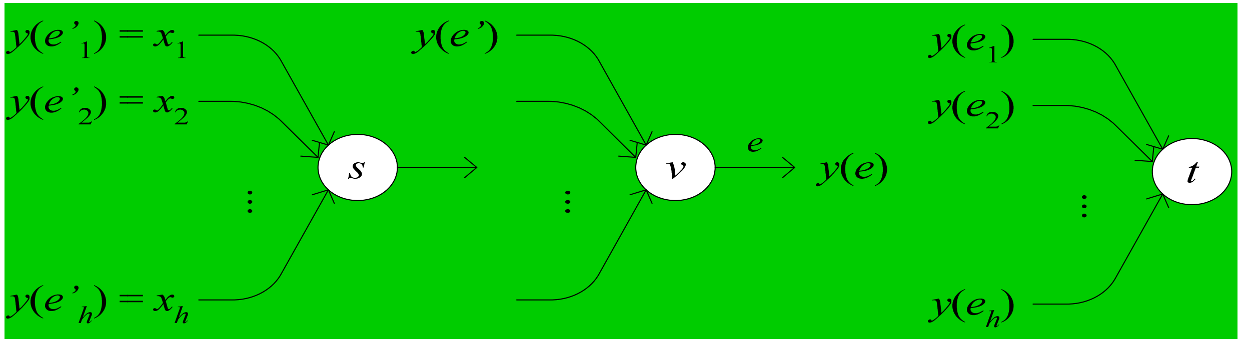
Algebraic Framework

- Graph (V,E) having unit capacity edges
- Sender s in V, set of receivers $T=\{t,\ldots\}$ in V
- Multicast capacity $h = \min_{t \in T}$ MaxFlow(s,t)



- $y(e) = \sum_{e'} \beta_e(e')\, y(e')$
- $\beta(e) = [\beta_e(e')]_{e'}$ is *local encoding vector*

Global Encoding Vectors



- By induction $y(e) = \sum^h{}_{i=1}\ g_i(e)\ x_i$
- $\mathbf{g}(e) = [g_1(e),\ldots,g_h(e)]$ is *global encoding vector*
- Receiver t can recover $x_1,\ldots,x_h$ from

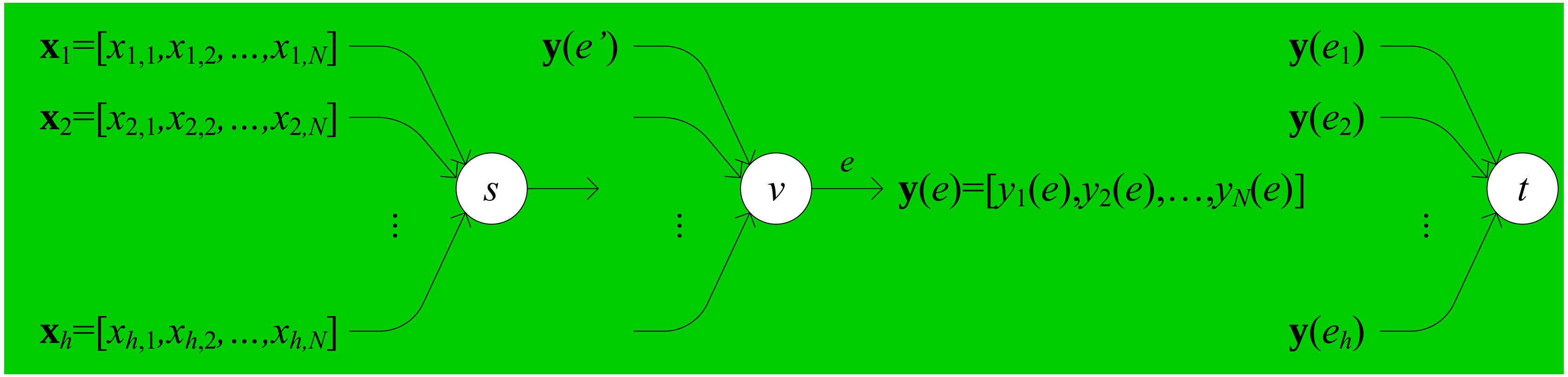
$$\begin{bmatrix} y(e_1) \\ \vdots \\ y(e_h) \end{bmatrix} = \begin{bmatrix} g_1(e_1) & \cdots & g_h(e_1) \\ \vdots & \ddots & \vdots \\ g_1(e_h) & \cdots & g_h(e_h) \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_h \end{bmatrix} = G_t \begin{bmatrix} x_1 \\ \vdots \\ x_h \end{bmatrix}$$

Invertibility of G_t

- G_t will be invertible with high probability if local encoding vectors are random and field size is sufficiently large
  - If field size = 2^{16} and $|E| = 2^8$ then G_t will be invertible w.p. $\geq 1-2^{-8} = 0.996$

[Ho, Koetter, Médard, Karger, and Effros; ISIT 2003]
[Jaggi, Sanders, Chou, Effros, Egner, Jain, and Tolhuizen; Trans IT 2005]

Packetization



- Internet: MTU size typically ≈ 1400^+ bytes
- $\mathbf{y}(e) = \sum_{e'} \beta_e(e')\,\mathbf{y}(e') = \sum_{i=1}^{h} g_i(e)\,\mathbf{x}_i$ s.t.

$$\begin{bmatrix} \mathbf{y}(e_1) \\ \vdots \\ \mathbf{y}(e_h) \end{bmatrix} = \begin{bmatrix} y_1(e_1) & y_2(e_1) & \cdots & y_N(e_1) \\ \vdots & \vdots & & \vdots \\ y_1(e_h) & y_2(e_h) & \cdots & y_N(e_h) \end{bmatrix} = G_t \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,N} \\ \vdots & \vdots & & \vdots \\ x_{h,1} & x_{h,2} & \cdots & x_{h,N} \end{bmatrix}$$

Packet Header

- Include *within each packet* on edge e
  $\mathbf{g}(e) = \sum_{e'} \beta_e(e')\, \mathbf{g}(e')$; $\mathbf{y}(e) = \sum_{e'} \beta_e(e')\, \mathbf{y}(e')$
- Can be accomplished by prefixing i th unit vector to i th source vector $\mathbf{x}_i$, $i=1,\ldots,h$

$$\begin{bmatrix} g_1(e_1) & \cdots & g_h(e_1) & y_1(e_1) & y_2(e_1) & \cdots & y_N(e_1) \\ \vdots & \ddots & \vdots & \vdots & \vdots & & \vdots \\ g_1(e_h) & \cdots & g_h(e_h) & y_1(e_h) & y_2(e_h) & \cdots & y_N(e_h) \end{bmatrix} = G_t \begin{bmatrix} 1 & & 0 & x_{1,1} & x_{1,2} & \cdots & x_{1,N} \\ & \ddots & & \vdots & \vdots & & \vdots \\ 0 & & 1 & x_{h,h} & x_{h,2} & \cdots & x_{h,N} \end{bmatrix}$$

- Then global encoding vectors needed to invert the code at any receiver can be found in the received packets themselves!

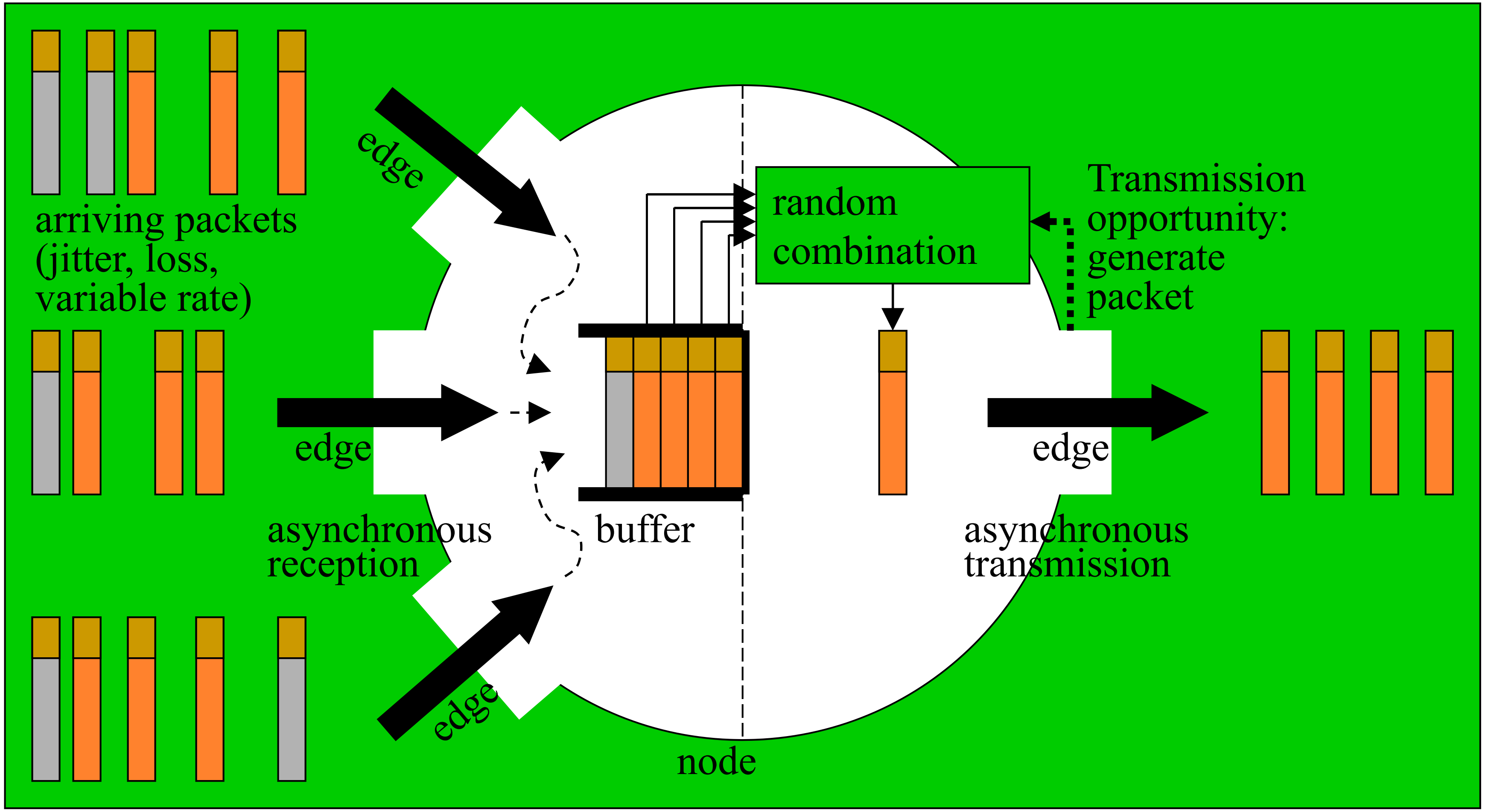
Header Cost vs. Benefit

- Cost:
  - Overhead of transmitting h extra symbols per packet; if $h = 50$ and field size $= 2^8$, then overhead ≈ $50/1400 \approx 3\%$
- Benefit:
  - Receivers can decode even if
    - Network topology & encoding functions unknown
    - Nodes & edges added & removed in ad hoc way
    - Packet loss, node & link failures w/ unknown locations
    - Local encoding vectors are time-varying & random

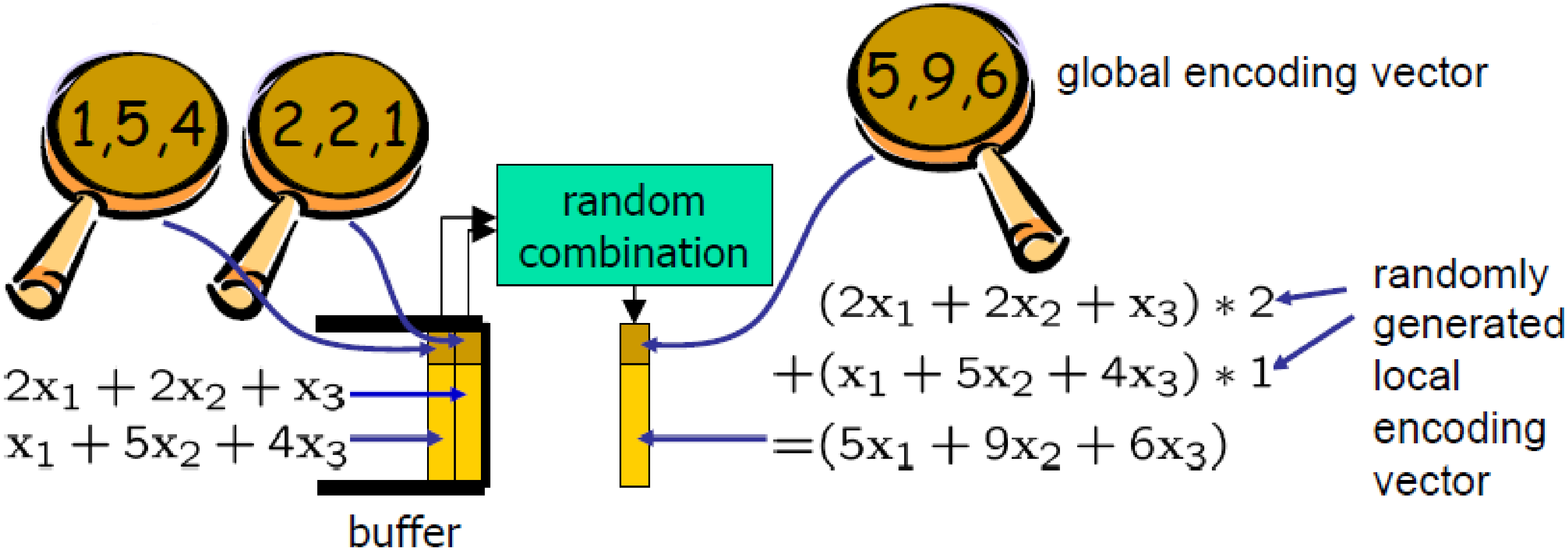
Asynchronous Communication

- In real networks
  - Packets on "unit capacity" edges between each pair of nodes are grouped and carried sequentially
  - Separate edges $\rightarrow$ separate prop & queuing delays
  - Number of packets per unit time on edge varies
    - Loss, congestion, competing traffic, rounding
- Need to synchronize
  - All packets related to same source vectors $\mathbf{x}_1, \ldots, \mathbf{x}_h$ are in same <u>generation</u>; h is <u>generation size</u>
  - All packets in same generation tagged with same <u>generation number</u>; one byte (mod 256) sufficient

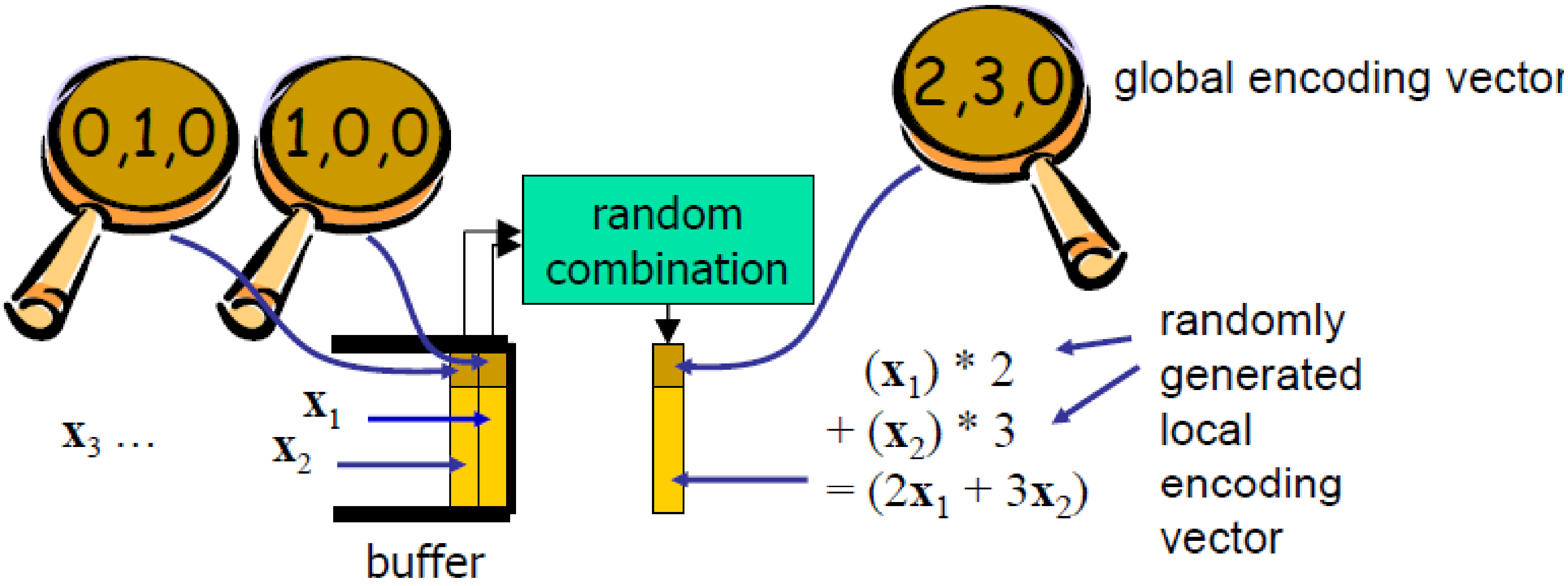
Buffering


edge
arriving packets
(jitter, loss,
variable rate)
random
combination
Transmission
opportunity:
generate
packet
edge
edge
asynchronous
reception
buffer
asynchronous
transmission
edge
node

At an Intermediate Node

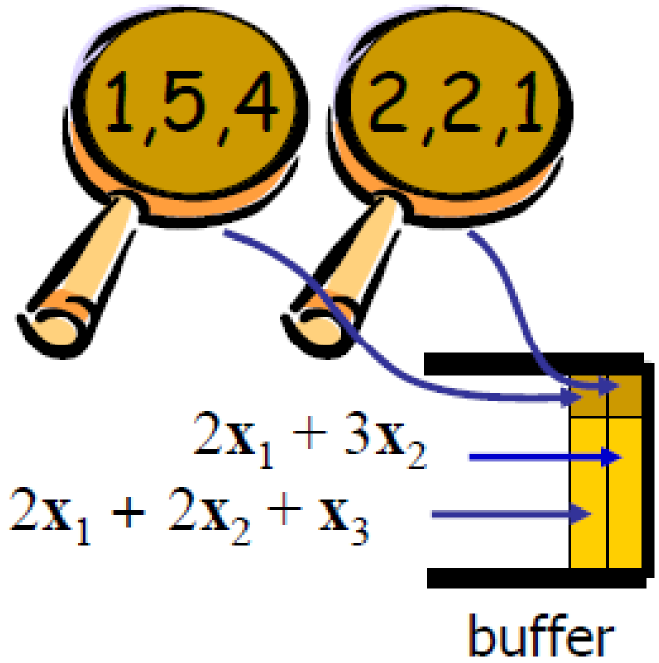


At the Source Node



At a Receiver Node

$$\mathbf{x}_1 + 5\mathbf{x}_2 + 4\mathbf{x}_3 \quad \ldots$$

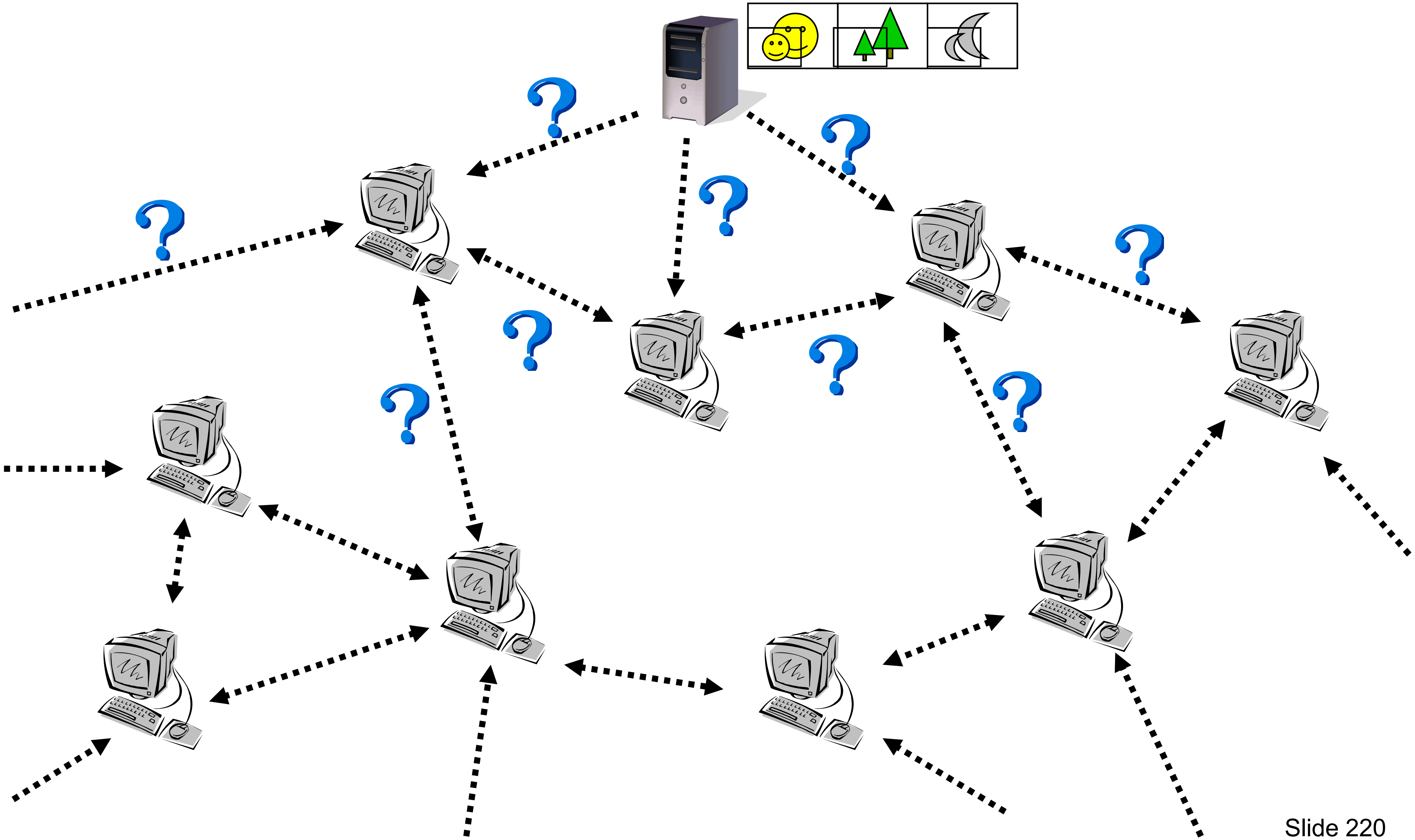


Application Scenario

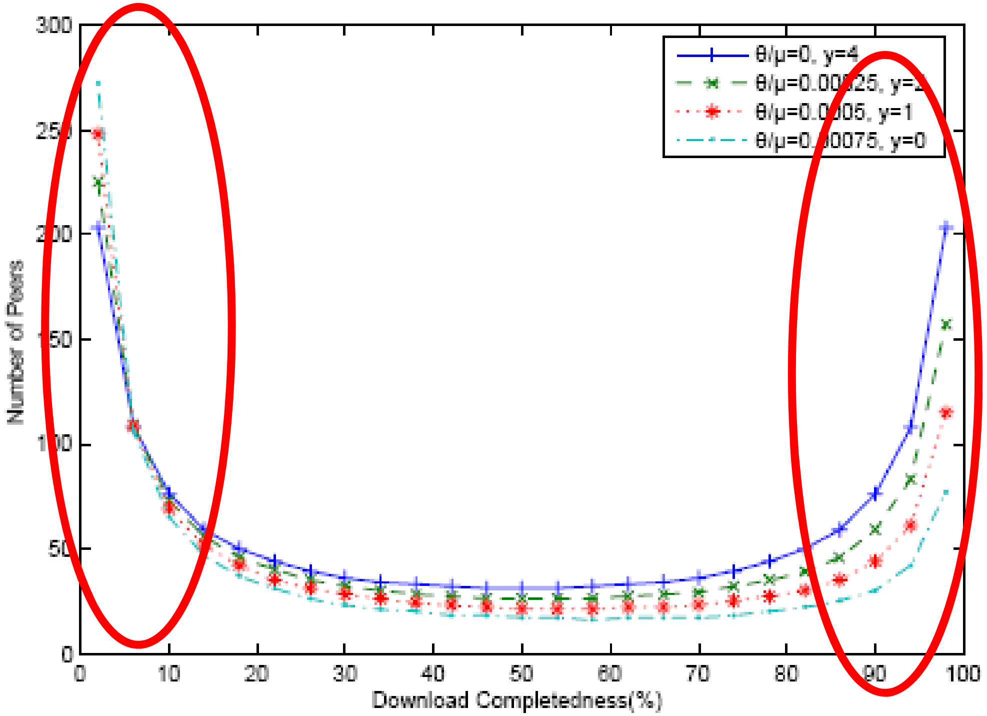
- File sharing – Avalanch [Gkantsidis-Rodriguez 05]

- Video-on-demand – UUSee [Liu-Wu-Li-Zhao 10]

File Swarm = Block Scheduling



System's progress in current File Swarming systems

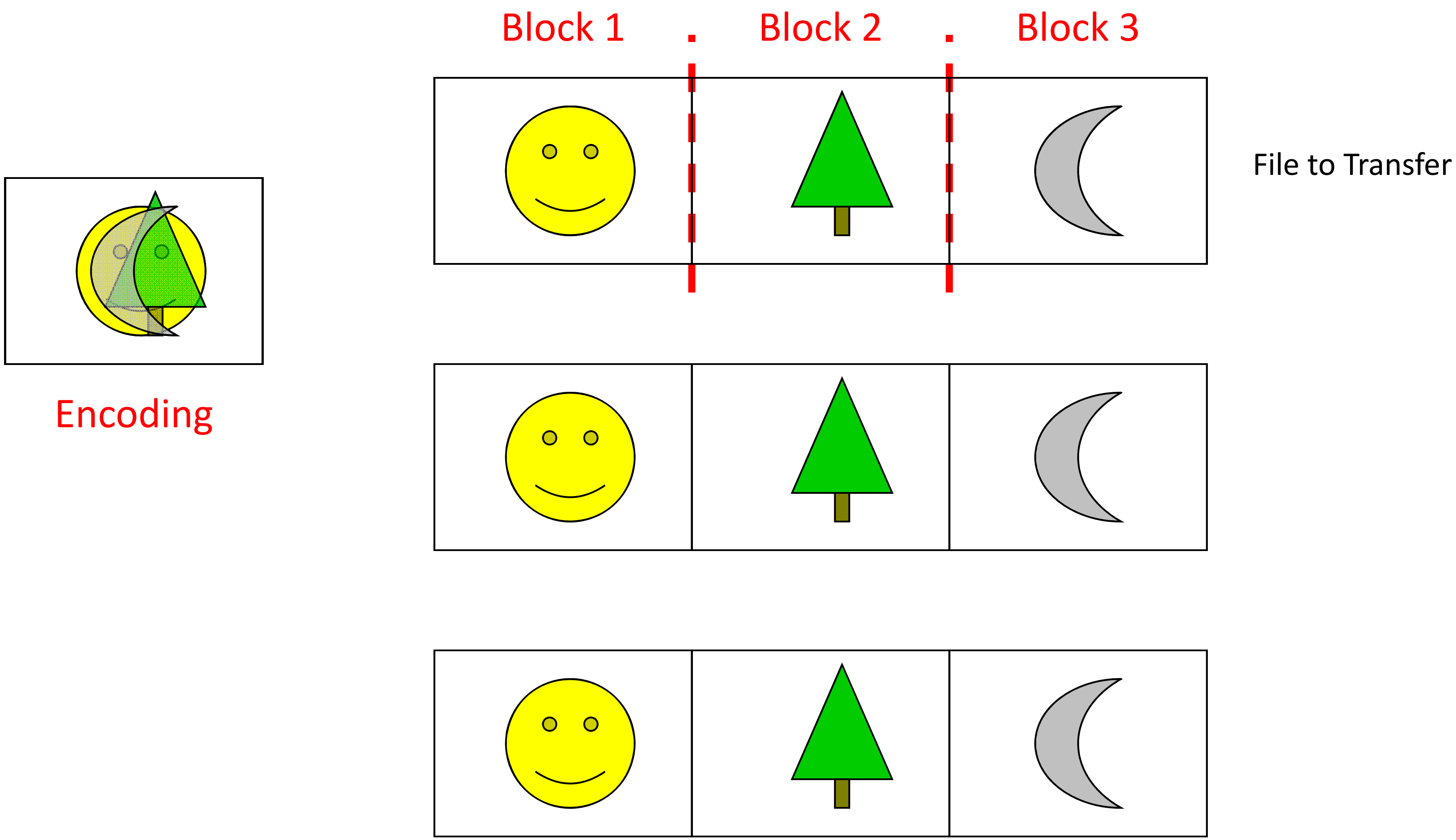


(From Tian et al., Infocom'06)

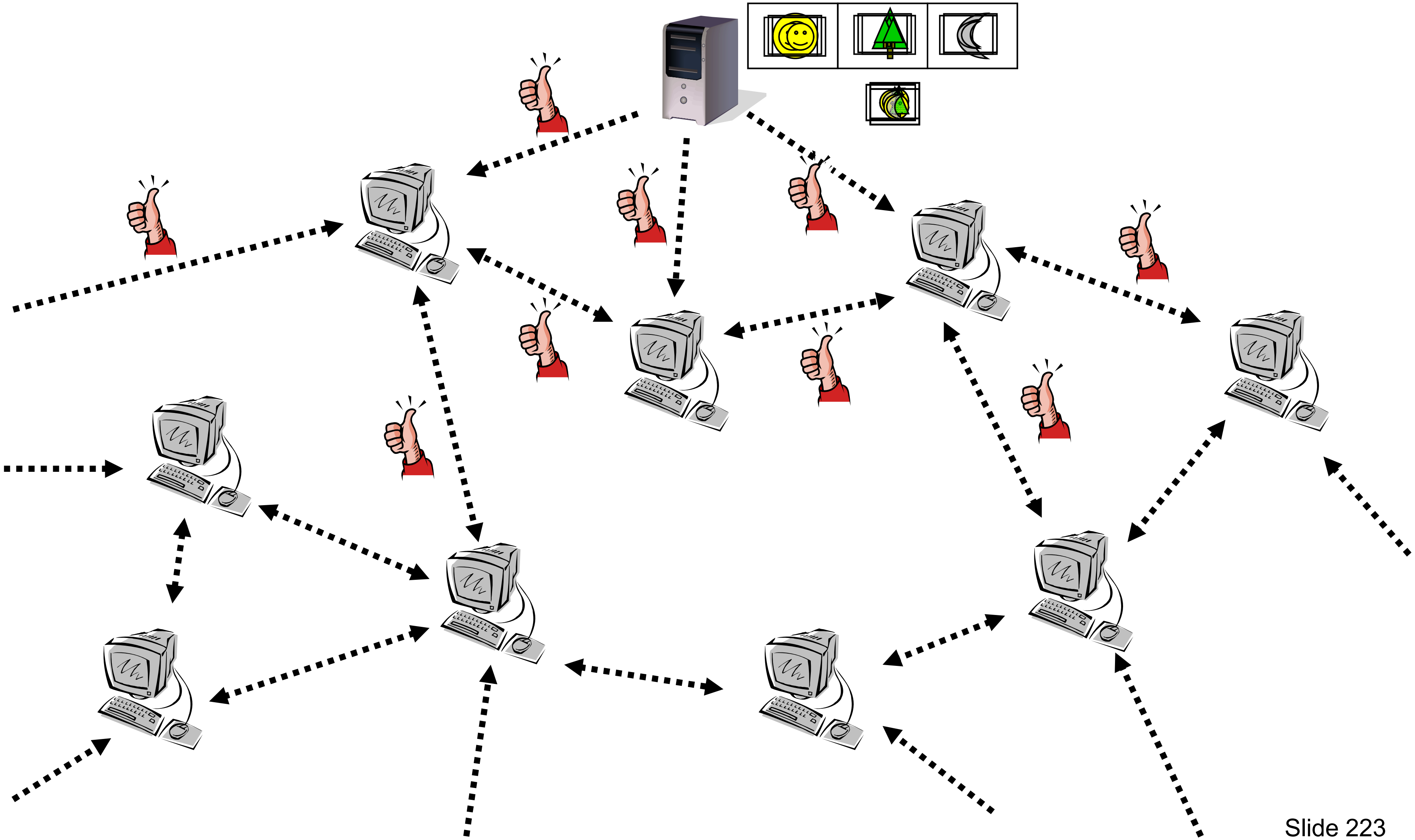
A lot of time spent at the beginning and finish of download:

- Beginning of download: finding good blocks to exchange
- End of download: discovering the last missing blocks

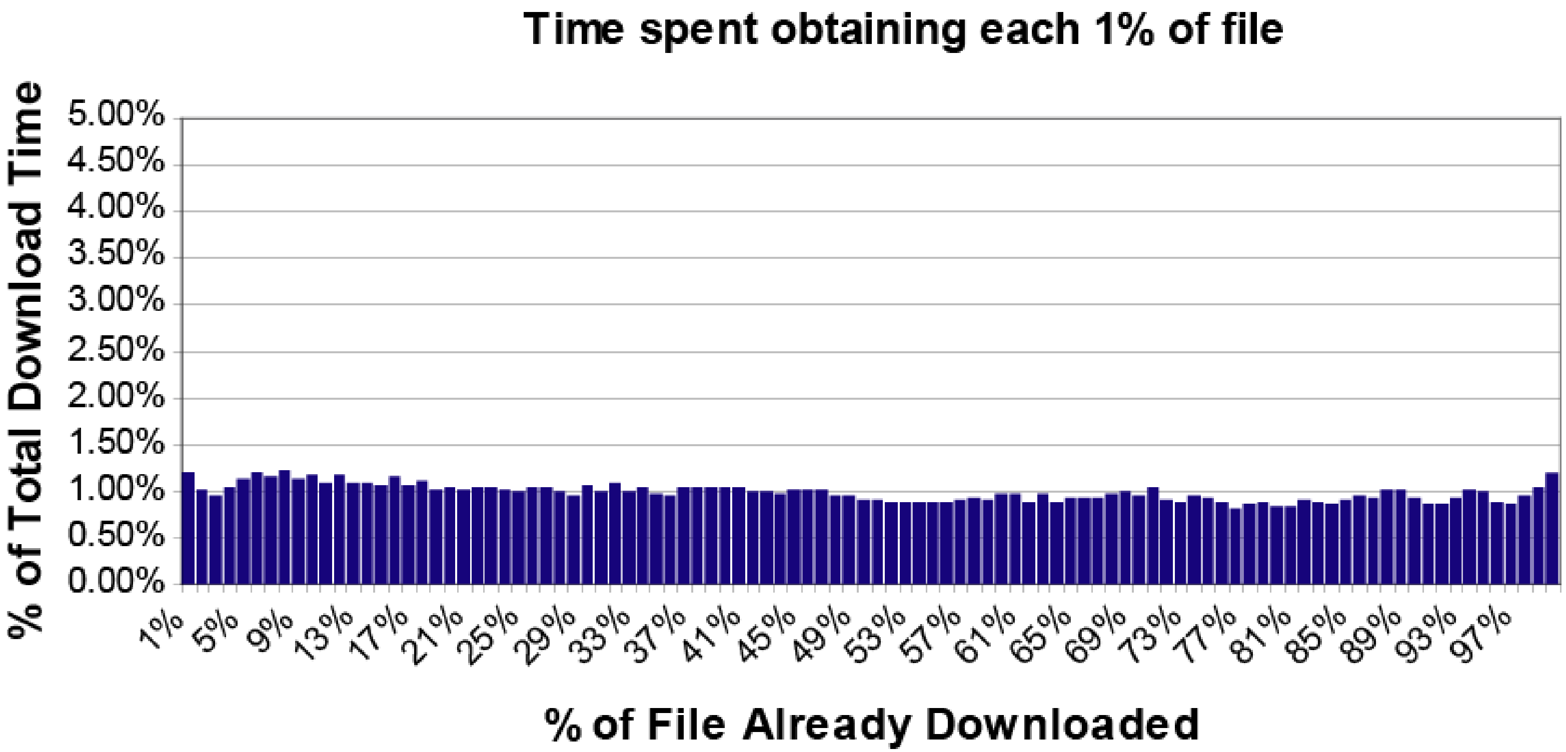
Network Coding Simplified



With Network Coding

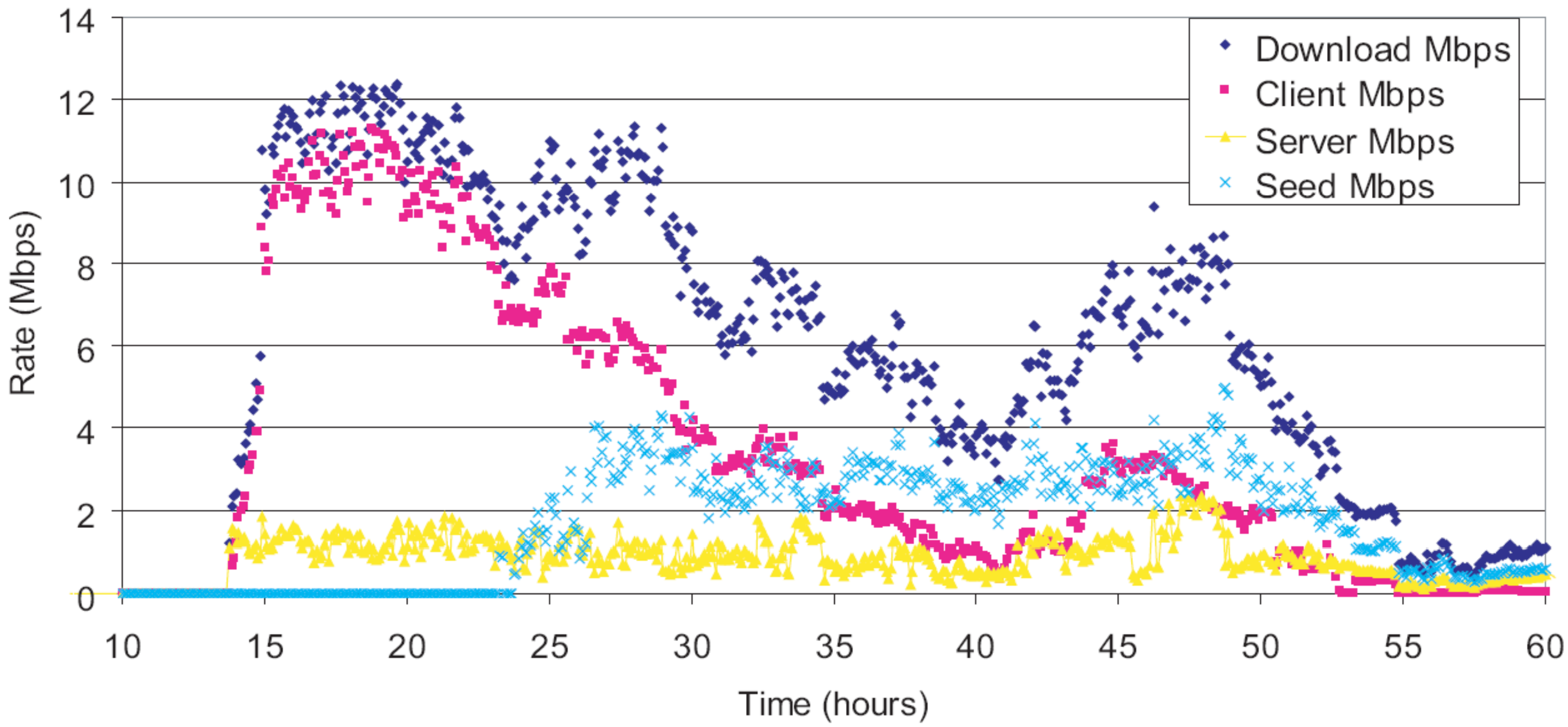


System's progress



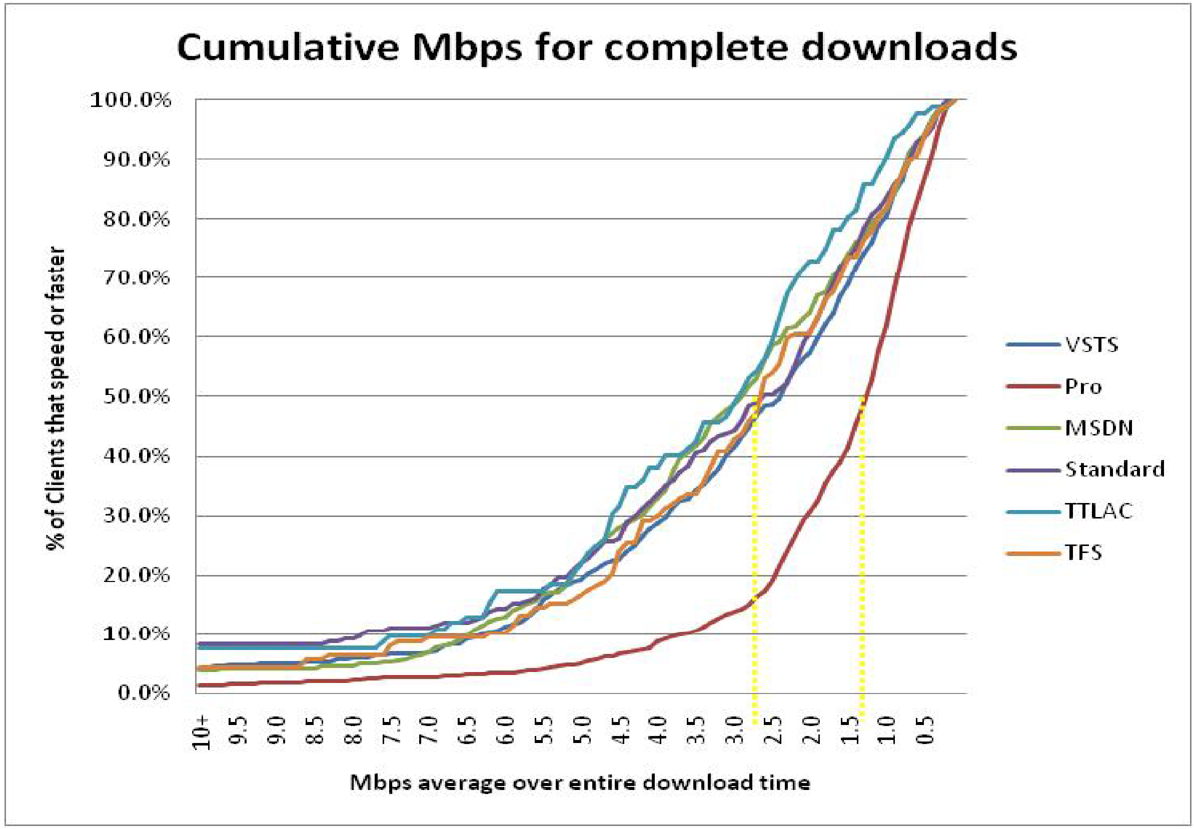
- Smooth download progress:
  - No start-up delay
  - No last-block problem

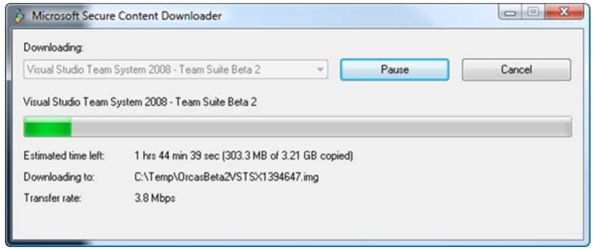
Bandwidth Contribution



- Easily withstands flash crowds
- Server contribution is fixed, Client contribution scales
- >10 fold savings in content provider’s bandwidth using peer-to-peer.

Results from distributing Visual Studio

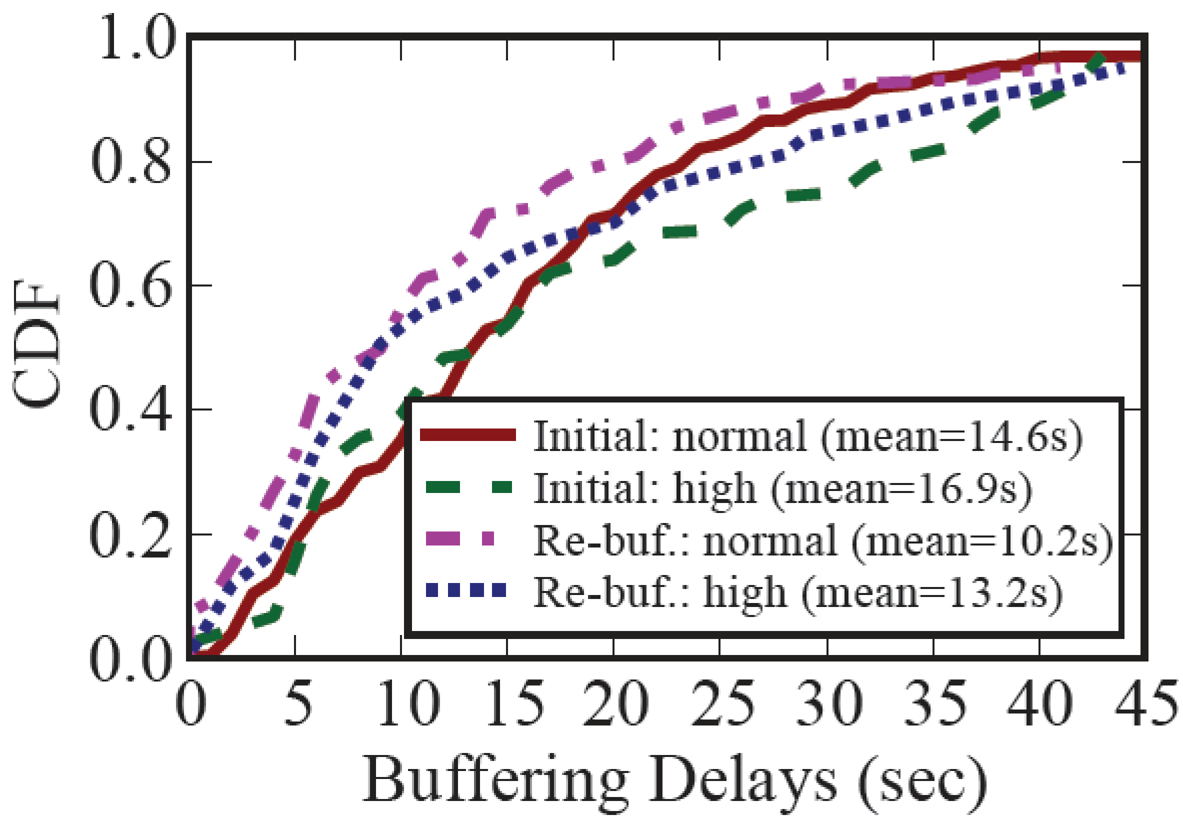




Data from distribution of beta versions of Visual Studio 2008 Beta (Nov'07)

Median speeds:
~1.5Mbps for VS Pro
~2.7Mbps for the others

Buffering Delay at A Random Seek in VoD [Liu-Wu-Li-Zhao 10]



10-17 seconds

Summary

- P2P applications are popular
- Throughput maximization of P2P systems is understood pretty well
- Delay minimization of P2P systems just starts
- Understanding and exploiting dynamics in P2P systems is still under-explored
- Network coding reduces the scheduling complexity in P2P significantly

- Will P2P become a service infrastructure?

Acknowledgement (slides material)

- Baochun Li (University of Toronto)
- Philip A. Chou (Microsoft Research)
- Joe Jiang (Princeton University)
- Yong Liu (Polytech University)
- Keith Ross (Polytech University)
- Yunnan Wu (Facebook)
- Shao Liu (Microsoft)
- Taoyu Li (Tsinghua University)
- Dah-ming Chiu (The Chinese University of Hong Kong)
- Laurent Massoulié (Thomson Technology Paris Laboratory)
- Di Wu (Sun Yat-Sen University)
- Xiaojun Hei (Huazhong University of Science and Technology)
- Yang Richard Yang (Yale University)

Thank You!

Questions?