Coolstreaming: Design, Theory, and Practice

Susu Xie, Bo Li, *Senior Member, IEEE*

Gabriel Y. Keung, and Xinyan Zhang, *Student Member, IEEE*

IEEE TRANSACTIONS ON MULTIMEDIA
DECEMBER 2007

1. Introduction
2. Detail of Cool Streaming
3. System Dynamics
4. Results and Discussions
5. Conclusion

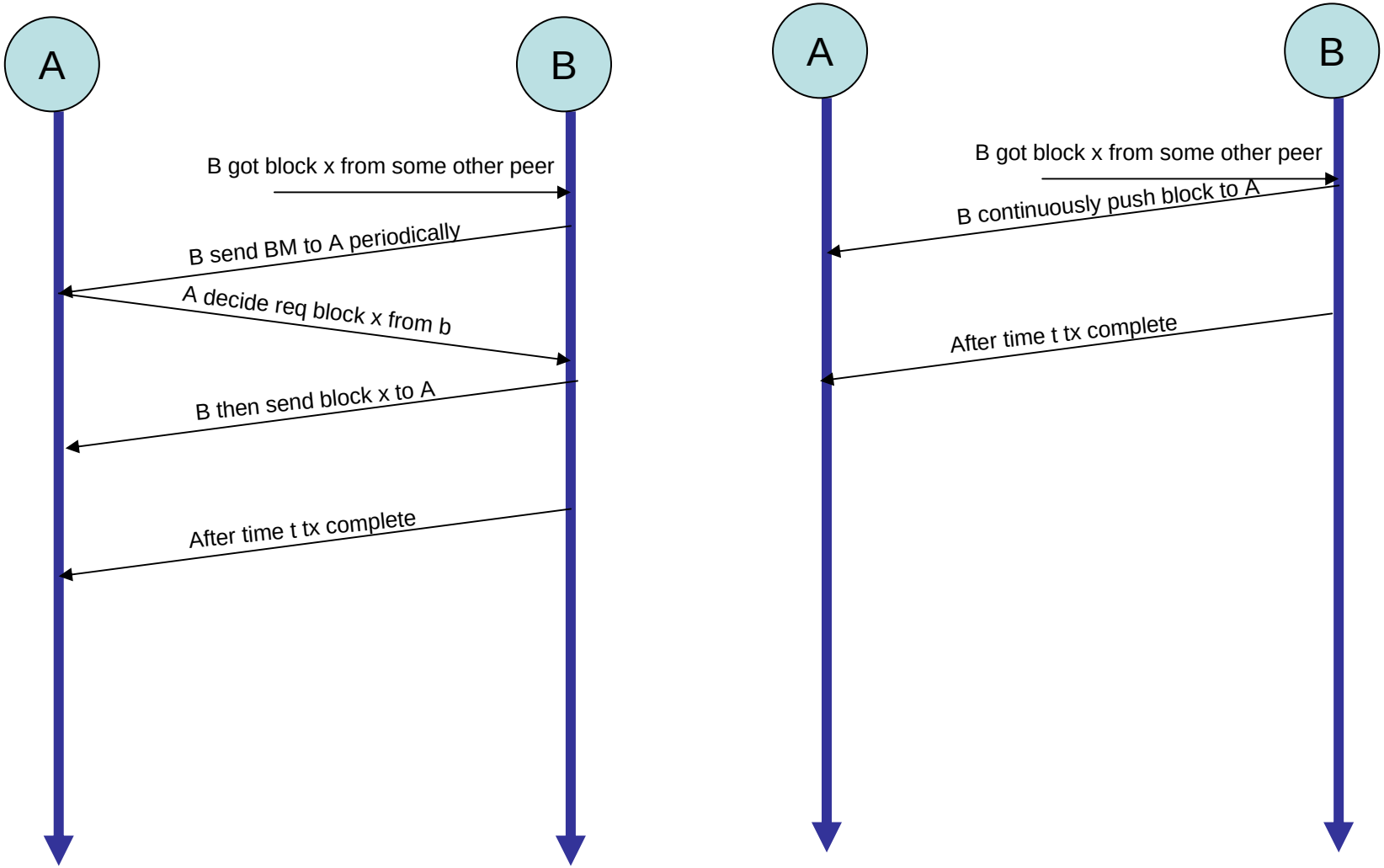
Introduction

- Coolstreaming represented one of the earliest large-scale P2P video streaming experiments [4](Infocom 2005), [6](MMSP 2005)
  - has been widely referenced in the community as the benchmark (Google entries tops 300 000)
  - a P2P-based live streaming system has the potential to scale

Enhanced

- 1) Since its first release, while keeping the random partner selection, we have enhanced the system in nearly all aspects
  - initial system adopted a simple pull-based
  - based on content availability information exchange using buffer-map
  - per block overhead
  - more importantly, it results in a longer delay in retrieving the video content
- now implemented a hybrid pull and push mechanism
  - pushed by a parent node to a child node
  - except for the first block (pull)

Pure pull delay



Enhanced

- 2) a novel multiple substream scheme is implemented
  - essentially enables multisource and multipath delivery for video streams
  - Observed from the results
    - not only enhances the video playback quality
    - but also improves the effectiveness against system dynamics
- 3) the gossip protocol was enhanced to handle the push function
- 4) the buffer management and scheduling schemes are redesigned to deal with the dissemination of multiple substreams

1. Introduction
2. Detail of Cool Streaming
3. System Dynamics
4. Results and Discussions
5. Conclusion

Original CS

- The original Coolstreaming system was developed in early 2004
- Since the first release, it has attracted millions of downloads worldwide
- The peak concurrent users reached over 80 000 with an average bit rate of 400 Kbps with clear global presence [6].

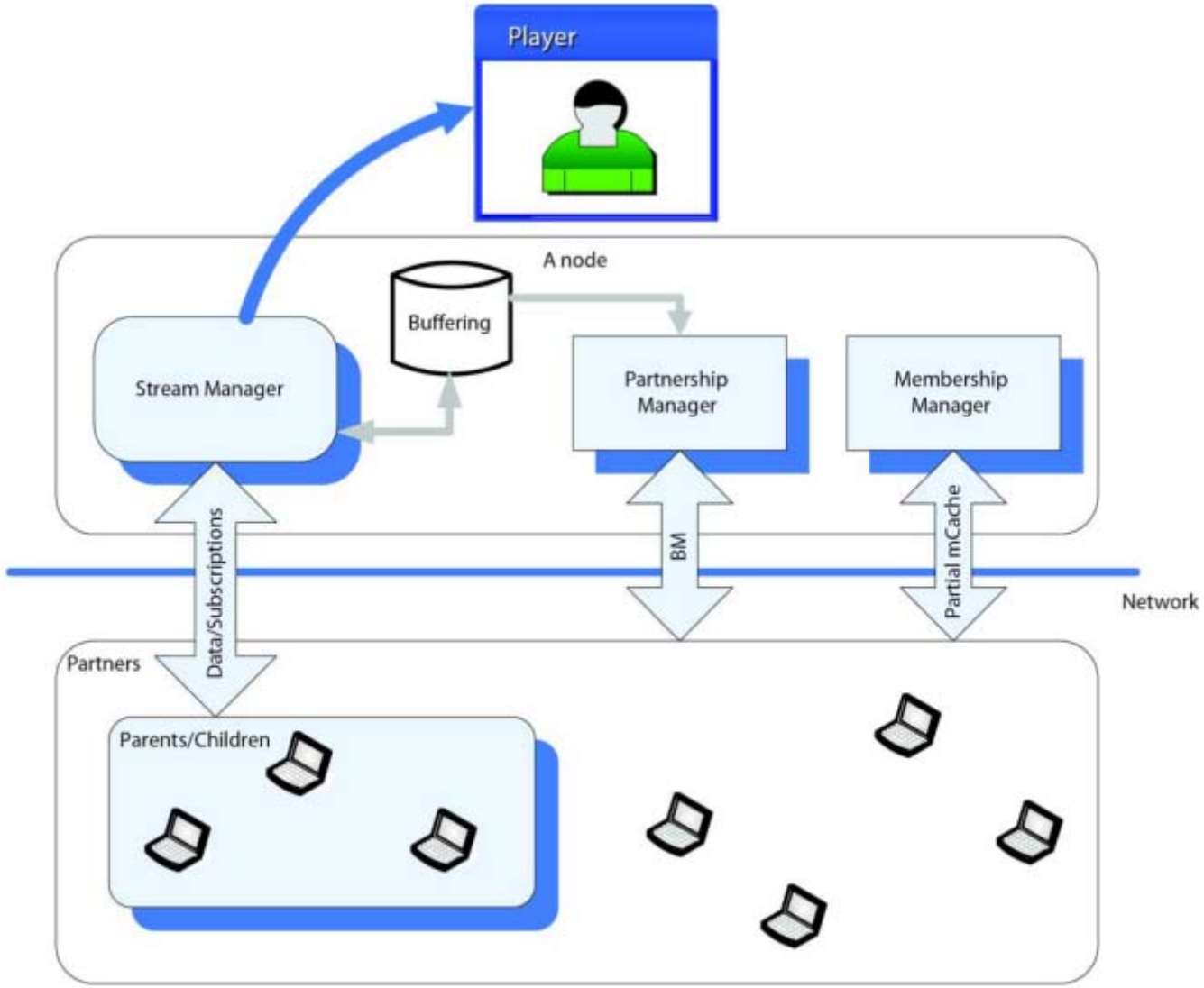


Fig. 1. Coolstreaming system diagram.

Multiple Substreams

- The video stream
  - divided into *blocks* with equal sizes
  - each block is assigned a sequence number
- a streaming and TCP for transmissions
  - the sequence number also serves as a timestamp
  - combine and reorder the blocks after reception

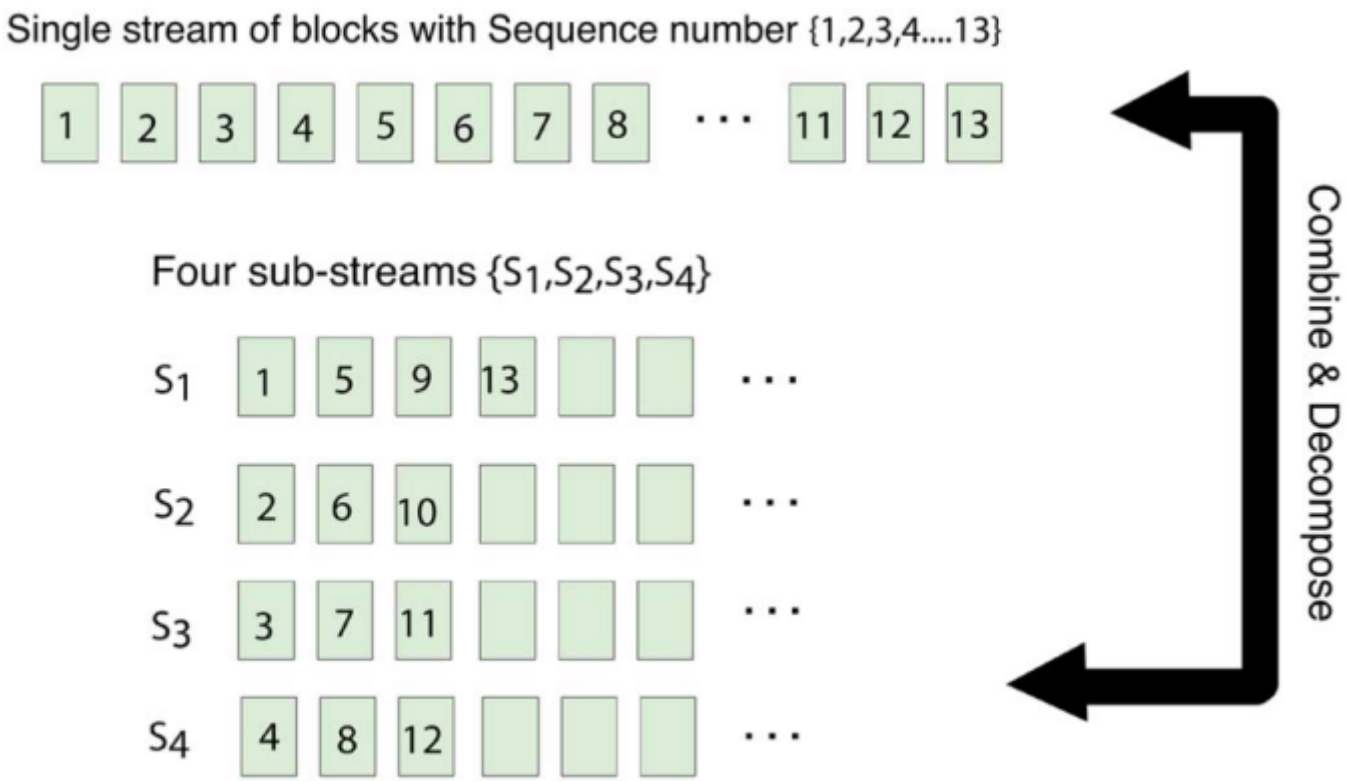


Fig. 2. Example of stream decomposition.

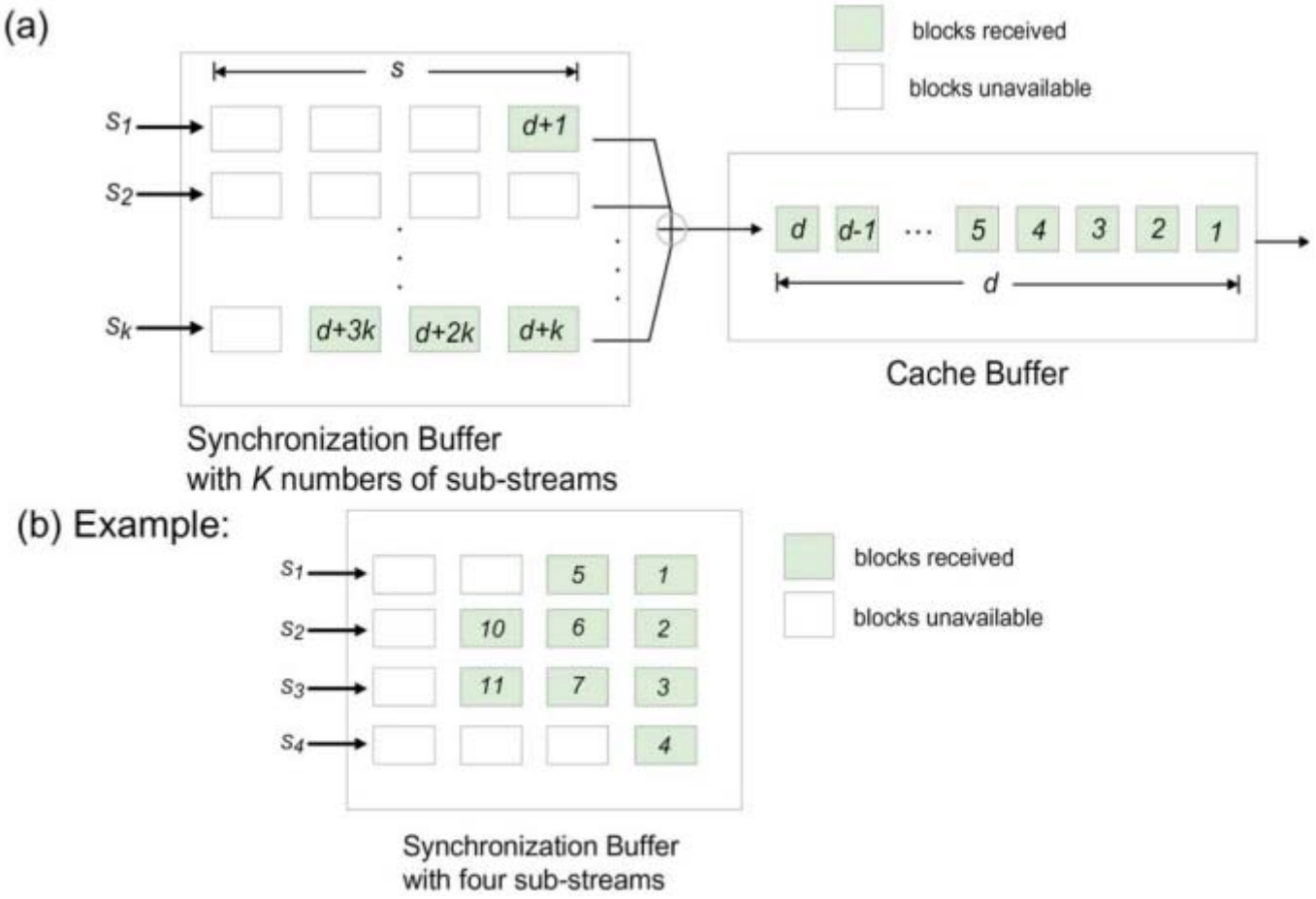


Fig. 3. (a) Structure of buffers in a node. (b) An example of combination process in Synchronization buffer.

Gossip

- key factors contributing to the success in P2P filesharing applications
  - adoption of the gossip concept
  - a node can request different small chunks of file content from different nodes
- This achieves significantly higher efficiency compared to other traditional systems [20]
- adopted in Coolstreaming
  - a video stream is divided into multiple *substreams*
  - nodes could subscribe to different substreams from different partners

2 relationships

- *Partnership* is different from *parent-children relationship*
- Partnership
  - Established on TCP
  - exchange block availability information
- parent-children relationship
  - a node (the child) is actually receiving substream(s) from another node (the parent)
  - It is common in CS that a parent node is sending multiple substreams to a child node
  - Dp, total number of substreams sent by node
  - A stream is decomposed into K substreams

Buffering

- buffer map (BM)
  - represent the availability of the latest blocks of different substreams in the buffer
  - exchanged periodically among partners to determine which substream to subscribe to
  - represented by a series of 2K-tuples
  - first K-tuples records the seq # of the latest received block
  - second of K-tuples represents the subscription of substreams from the partner
  - A send {1,1,0,0,…..,0} to B for requesting 1st and 2nd substreams

Overlay

- A *membership manager* is an essential
- Each node in the system has an unique identifier and maintains a membership cache (mCache)
- mCache containing a partial list of the currently active nodes in the system
- Each node in the mCache is considered as a *member*
- The system consists of an origin or source node, a boot-strap node and member nodes
  - *boot-strap* node serves as the entry point (deliver mCache to new join node, like tracker in BT)

Overlay

- The gossip protocol is used for overlay construction
- widely used in BitTorrent and other P2P systems
  - achieves excellent resilience against random failures
  - enables decentralized operation
  - a newly joined node contacts the boot-strap node
- boot-strap node performs two simple operations
  - providing a randomized list of the currently active peer nodes to the newly joined node
  - possibly updating its mCache by including the newly joined node

Overlay

- Based the mCache, the node randomly selects a few nodes to establish TCP connections, i.e., partnership.
- Once the partnership is established, exchange their mCache contents
  - only executed when the partnership is first established
- mCache is bounded by a system parameter M
  - exchange of mCache can possibly cause a node to remove some entries from its mCache
  - each node can periodically update its mCache entry to maintain a list of currently active nodes in the system
  - Once a node was recognized as disabled, it simply removes from mCache
  - Rather advertise immediately, thus will not cause flooding
  - Such info will be gradually fade out over a period of time
    - Periodical exchange mCache
    - Reduce the control messages

Content Delivery

- Coolstreaming adopts a hybrid *push and pull* scheme
- A node subscribes to a substream by connecting to one of its partners via a single request (pull) in BM
- The parent node will continue pushing all blocks in need of the substream to the child
- In CS, a parent node will not voluntarily drop a child node
- It is up to the children to dynamically monitor the in-coming connections and trigger any parent reselection if necessary

Req from ?

- the newly joined knows the availability of it's parent node
- It has to determine from which part of the stream should the newly joined node request
- This turns out to be a nontrivial problem
  - I think so……..
- Suppose the range of the blocks available in all its partners are from n to m
- Intuitively, the node should request from a block with sequence number somewhere in the middle
- If starting from m
  - the partner nodes might not have sufficient follow-up blocks to satisfy the continuity requirement for the video stream
- What if starting from n?
  - such blocks might no longer be available
  - It takes long to catch up with the current video stream

Tp

- Tp is a parameter which will be introduced later
- It is a boundary to decide if obsolete a parent
  - If the latest block of a parent is Tp blocks behind all partner's latest block
- The node finds the largest sequence number m of the received blocks among its partners
- The newly joined node will start to subscribe from a block shifted by a system parameter Tp

1. Introduction
2. Detail of Cool Streaming
3. System Dynamics
4. Results and Discussions
5. Conclusion

Peer Adaptation

- It is up to the child to decide if obsolete a parent
- The child constantly monitor the status of the on-going substream transmissions
- Peer Adaptation is the action that retires current parent and reselects new parent
- The questions are what triggers this adaptation, i.e., how to detect possible congestion or churns, and how to reselect new parent(s)

2 thresholds

- {Ts, Tp}
  - related to the sequence number of blocks for different substreams in each node (say, node A)
- Ts
  - can be interpreted as the threshold of the maximum sequence number deviation allowed between the latest received blocks in any two substreams in node A
- Tp
  - is defined as the threshold of the maximum sequence number deviation of the latest received blocks between the partners and the parents of node A
- $H_{si,A}$ as the sequence number of the latest received block for substream Si at node A

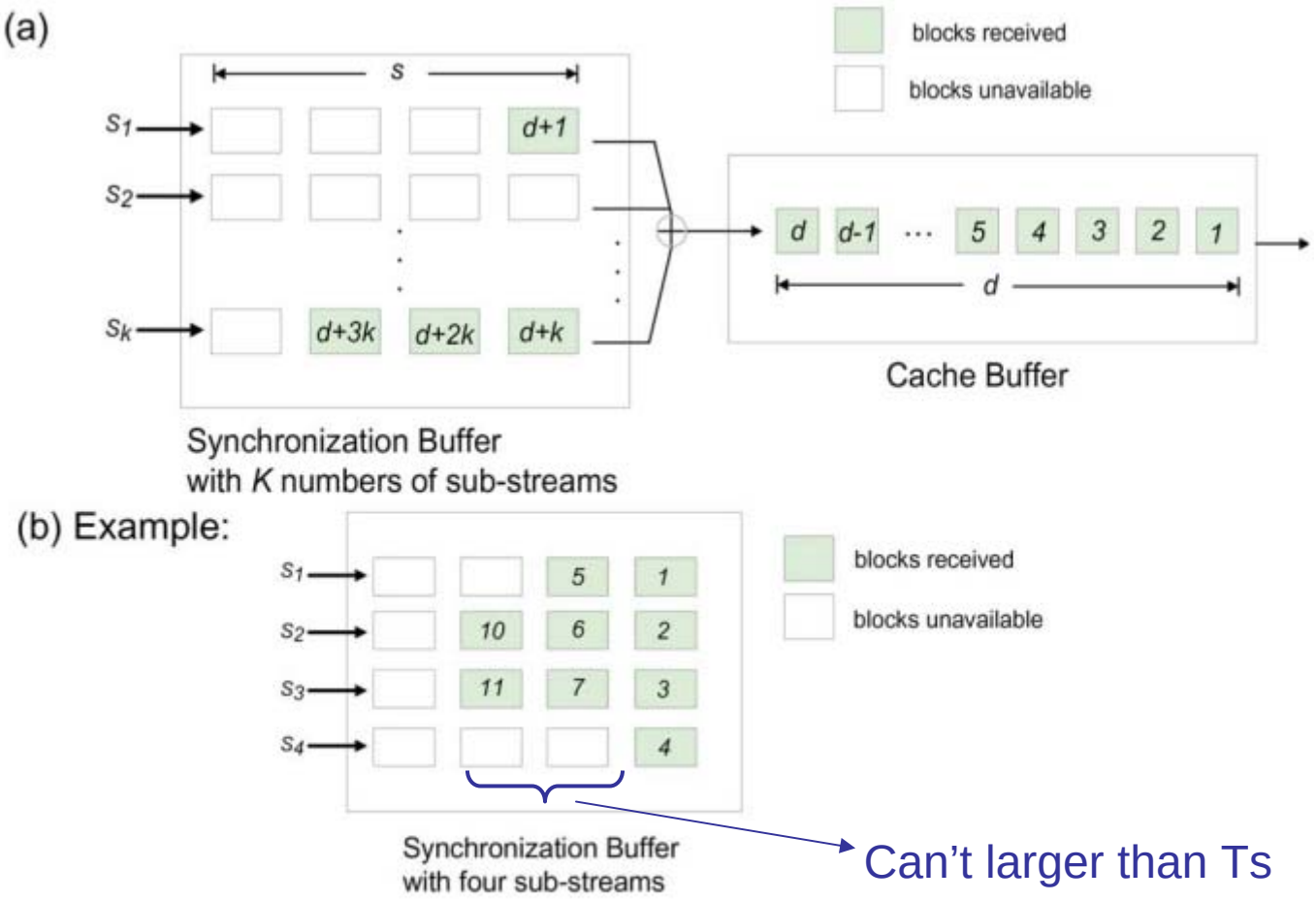


Fig. 3. (a) Structure of buffers in a node. (b) An example of combination process in Synchronization buffer.

2 inequalities

- (1) is compared within A
- (2) is compared to all A's partners
- If one ss violate (1), it means that this ss lead or delay some ss at least Ts blocks
- If one ss violate (2), it means that this ss lag at least Tp blocks to the fastest one within A's partners

$$\max\{|H_{S_i,A} - H_{S_j,p}| : i \leq K\} < T_s \quad (1)$$

$$\max\{H_{S_i,q} : i \leq K, q \in \text{parnters}\} - H_{S_j,p} < T_p. \quad (2)$$

Peer Adaptation

- (1) an indication for either congestion or insufficient upload capacity for this ss
- If (2) does not hold, it implies that the parent node is considerably lagging behind when comparing to at least one of the partners, which currently is not a parent node for the given node
- Both violations of (1) or (2) also trigger peer adaptation
- The new selected partner must satisfy the two inequalities
- If more than one qualified peers, choose randomly

Parent

- A parent node, however, will always accept requests and it will simply push out all blocks of a substream in need to the requesting node
- Apparently such a peer adaptation can potentially cause stream disruption and instability of the overlay topology (possible chain reaction)
- A cool-down timer is introduced to confine nodes to perform peer adaptation once only within a cool-down period of time Ta

chain reaction

- The only restriction of parent node is M
- This is because the parent will continue accepting new children as long as its total number of partners is less than M
- In this case, all children nodes have to compete for the insufficiently aggregated upload capacity from the parent
- We call this situation *peer competition*
  - eventually one or more nodes will lose and trigger peer adaptation.
- This causes a chain reaction of peer adaptations
- During the process, the inequalities can be violated (because Ta) and some temporary parents (lose competition) may be selected and abandoned before a capable parent is located

Peer Dynamics

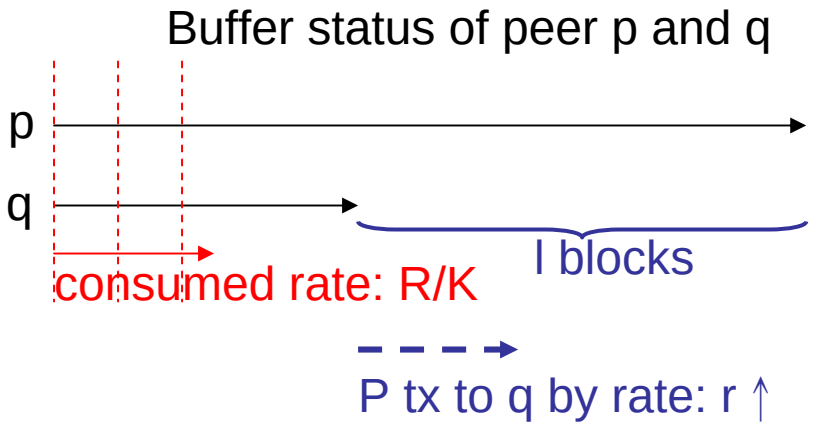
- The average bit rate required for one single ss transmission is R/K
  - R is the bit rate of live video stream
- Consider the case that node (the parent) is pushing data to node (the child)
- Suppose p is capable of providing upload bandwidth $r_{\uparrow} > R/K$
  - Q can eventually catch up with p in terms of the missing blocks from a substream
  - Refer to *catch up process*

Catch up time

- assume initially there are l blocks missing from node q comparing to node p
- then time $t_{\uparrow}$ for the catch up process can be easily computed by

$$r_{\uparrow} \cdot t_{\uparrow} = R/K \cdot t_{\uparrow} + l$$

$$t_{\uparrow} = \frac{l}{r_{\uparrow} - R/K}.$$



Time limit

- Suppose the parent of node p (grand father) is incapable of supporting its children nodes including p
- P loses the competition and if cannot find a new capable parent fast enough, its children (such as node q) can be stalled
- The time, denoted as $t_{\downarrow}$, for a child of parent to abandon as its parent, i.e., the ss obtained from node p in node q lags behind other ss beyond threshold Ts
- If p can find a new capable parent within time $t_{\downarrow}$, there is no need for node q to perform peer adaptation

Abandon time

- A node can still receive blocks from those temporary parents, suppose with an average bit rate $r_{\downarrow}$, and $r_{\downarrow}$ < R/K
- Then time $t_{\downarrow}$ can be computed by

$$t_{\downarrow} \cdot r_{\downarrow} + l = t_{\downarrow} \cdot R/K$$

$$t_{\downarrow} = \frac{l}{R/K - r_{\downarrow}}.$$

From saturated to unsaturated

- Suppose the ss degree of p is Dp when q is accepted as a child, and can satisfy (saturated) all its children before q's subscription
- After q is accepted, the upload bandwidth for each ss transmission of p decreases from R/K down to $r_{\downarrow}$, where

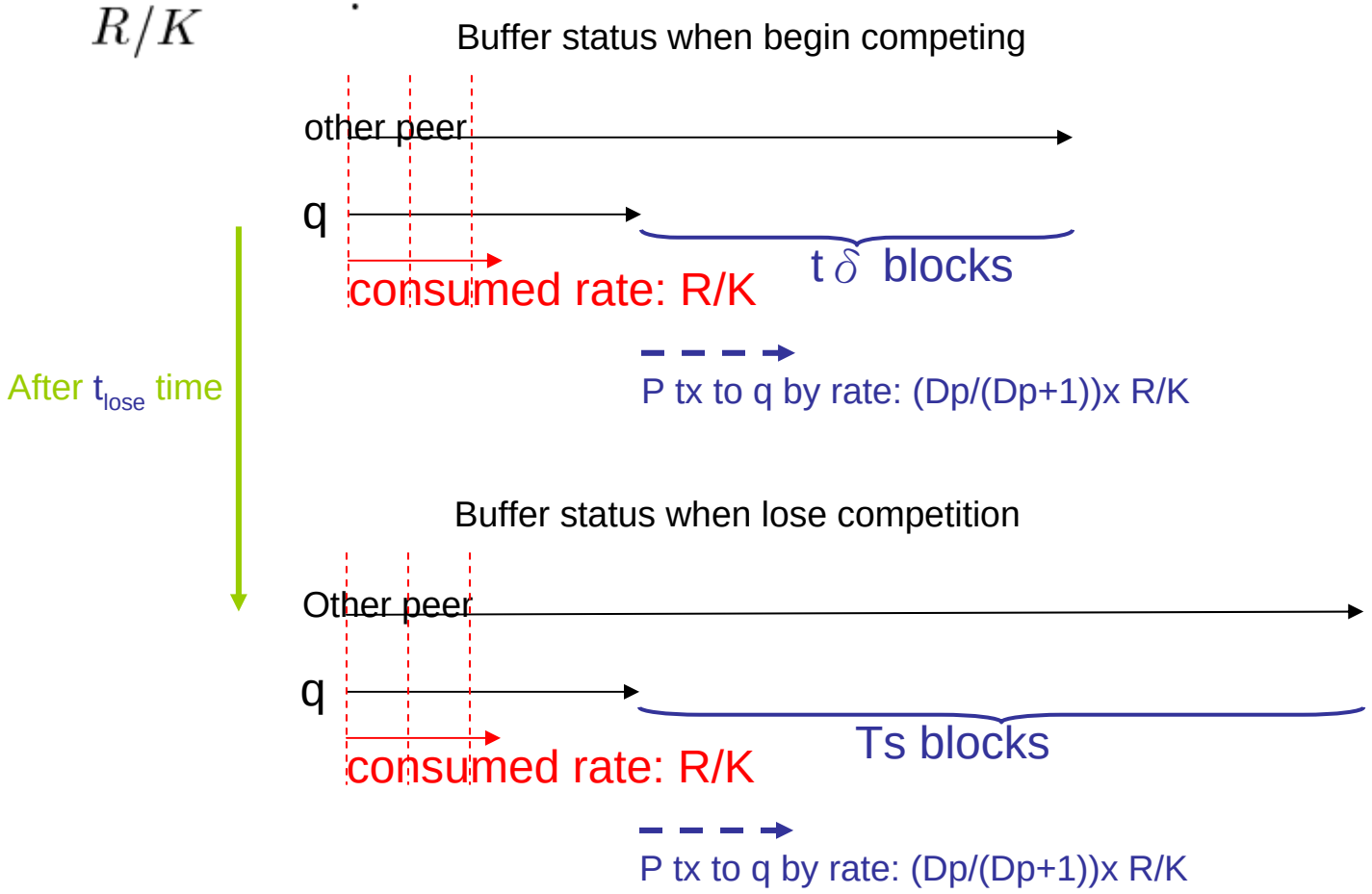
$$r_{\downarrow} = \frac{D_p}{D_p + 1} \cdot R/K.$$

Result of competition

- The result of competition depends on the buffer status of the children nodes at the beginning of the competition
  - When the ul rate of it's parent turn to unsaturated, the competition is begin
- Suppose it takes time t_{lose} for one of the children to lose the competition due to a subscribed substream lagging behind others from to t_{δ} in Ts unit of blocks, i.e., it violates Inequality (1).

$$R/K \cdot t_{\text{lose}} - \frac{D_p}{D_p+1} \cdot R/K \cdot t_{\text{lose}} = (T_s - t_\delta)$$

$$t_{\text{lose}} = \frac{(D_p+1)(T_s - t_\delta)}{R/K}.$$

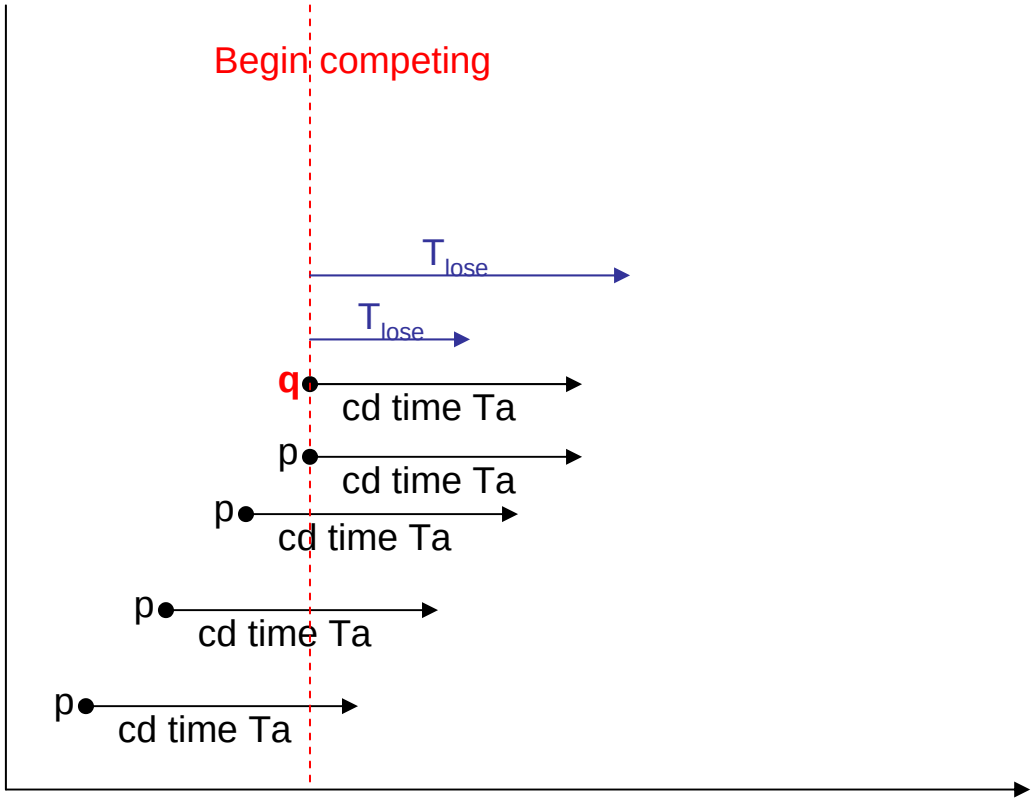


Probability for lose competition

- system confines nodes to perform peer adaptation once only in time period Ta
- Node q will unsubscribe from p if there is no other children of p losing the competition within Ta
- The probability for a child to lose the competition

$$P(t_{\text{lose}} \leq T_a) = P\left(\frac{(D_p+1)\cdot(T_s - t_\delta)}{R/K} \leq T_a\right)$$
$$= P\left(t_\delta \geq T_s - \frac{T_a \cdot R/K}{D_p+1}\right).$$

Competition



1. Introduction
2. Detail of Cool Streaming
3. System Dynamics
4. Results and Discussions
5. Conclusion

Results and Discussions

- In this section, we present the results obtained from a live streaming event using Coolstreaming on 27 September, 2006

4 kind of peers

- The CS system employs a lightweight protocol that is based only on the local information received from each user
- users can be classified as private or public users
- By checking whether we are successful to open various TCP connections, we can further classify users into the following four types
  - *Direct-connect*
    - Public ip and income/outgoing peers
  - *UPnP*
    - *Private ip* but income/outgoing peers
  - *NAT*
    - Private ip and outgoing peers only
  - *Firewall*
    - *Public ip but* outgoing peers only

Uneven upload distribution

- Use outgoing partners to illustrate the fact that it is only feasible for NAT or firewall users to initialize the partnership establishment, not the other way around
- However, once a NAT or firewall user establishes a partnership with another node, it can still upload video to other node whenever it is capable of doing so
- A NAT or firewall user can still become the parent for another peer node
- Specifically, 30% or so peer nodes in the overlay, i.e., nodes under UPnP and direct-connect, contribute more than 80% of the upload bandwidth

Uneven upload distribution

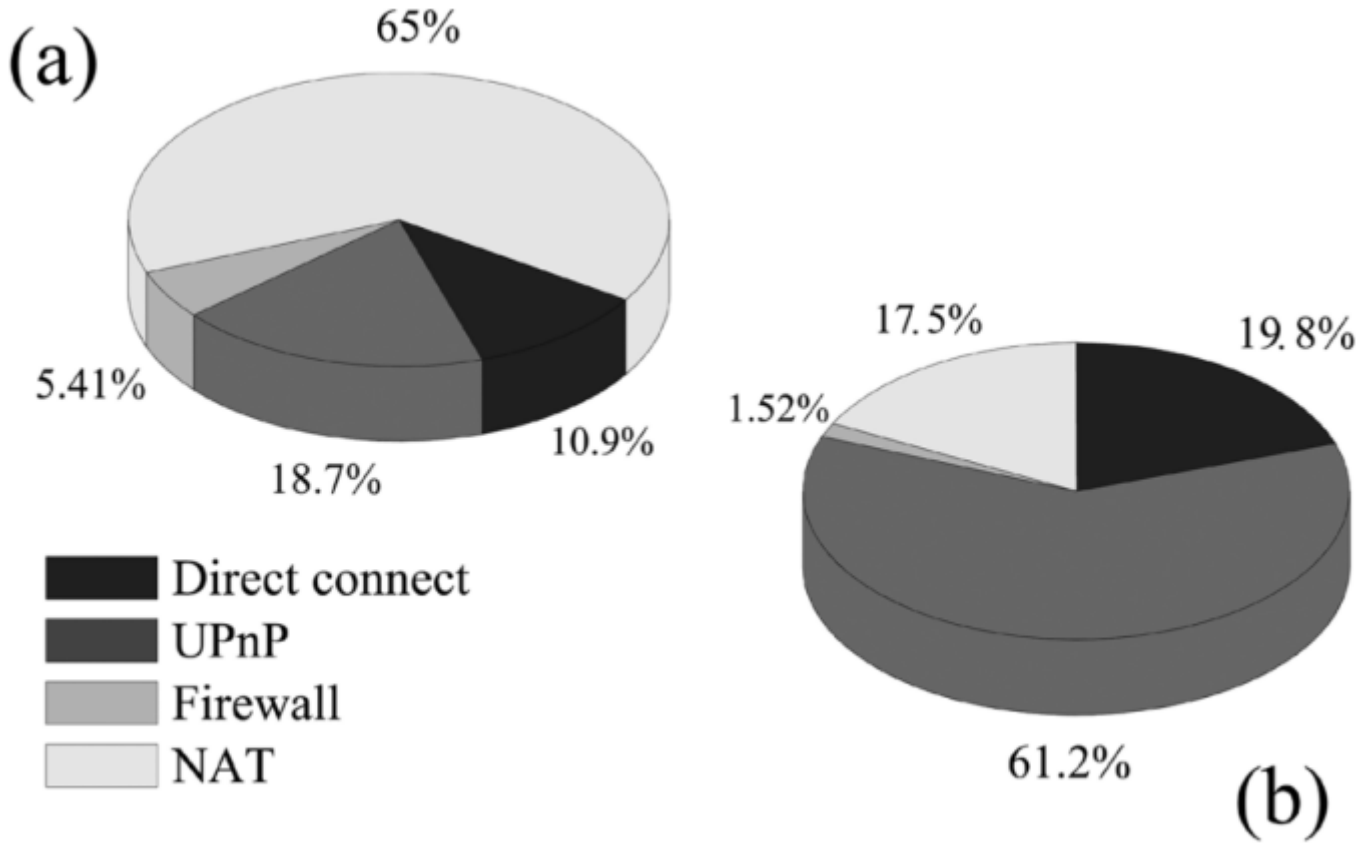


Fig. 4. (a) User Type Distribution; (b) User Contribution Distribution.

Convergence of the Overlay

- The overlay can be affected by three possibilities during a peer adaptation
- If node q become unstable and after performing of peer adaptation it find p
  - A–(*adaptation absorption*)
    - p has enough free upload bandwidth over R/K.
    - the overlay turns into a new stable state.
  - B–(*adaptation relay*)
    - P does not have sufficiently free upload bandwidth to serve q
    - Q compete and win, then another one perform peer adaption
  - C–(*adaptation amplification*)
    - Q can't find parent in time or continuously loses
    - Q' 's children will then become unstable
    - The number of unstable peers is amplified to Dq+1

Convergence of the Overlay

- We can model peer adaptation by a *continuous time branching process* [24]
- Since we are only concerned with whether the overlay will converge to a stable state eventually, we consider the corresponding *embedded generation process*, which counts the numbers of stable and unstable nodes in each generation
- To simplify our discussion, we assume q selects one parent when peer adaptation occurs
- Under this assumption, the reproduction rate ξ_q of an unstable peer and expected value of ξ_q(E(ξ_q)) can then be computed by

Convergence of the Overlay

- 1 = P(A) + P(B) + P(C)
- P(B) + P(C) = 1 – P(A)
- The expected value of ss degree in an overlay should equal to the total number of uploading ss by nodes over the population, which is approximately K (i.e., E(Dq) = K).

$$\begin{aligned} \xi_q &= 0 \cdot P(A) + P(B) + (D_q + 1) \cdot P(C) \\ &= 1 - P(A) + D_q \cdot P(C) \\ E(\xi_q) &= 1 - P(A) + E(D_q) \cdot P(C) \end{aligned}$$

Convergence of the Overlay

- Base on the theory of branching process [24], if is E(ξ_q) less than 1, the population of unstable peer eventually becomes extinct after several generation steps, which is called *subcritical* branching process

Theorem 1: An overlay will be of self-convergence $\Longleftrightarrow$ peer adaptation is a *subcritical* branching process, or $E(\xi) < 1$ in (7).

Proof of Theorem 1:

$$E(\xi_q) = 1 - P(A) + E(D_q) \cdot P(C) < 1$$
$$P(A) > E(D_q) \cdot P(C).$$

Then, the probability of event **C** happens at time $t_{\downarrow}$, which is the end of the chain reaction of peer adaptations, is

$$P(C) = [1 - P(A) - P(B)]^{\frac{t_{\downarrow}}{T_a}}.$$

From (4) and (5), we have

$$\frac{t_{\downarrow}}{T_a} = (D_p + 1) \cdot \frac{l}{T_a}. \tag{8}$$

Then the problem of overlay convergency becomes to be

$$E(\xi_p) = \frac{P(A)}{K \cdot [1 - P(A) - P(B)]^{(D_p+1) \cdot l/T_a}}. \tag{9}$$

$$t_{\downarrow} \cdot r_{\downarrow} + l = t_{\downarrow} \cdot R/K$$

$$t_{\downarrow} = \frac{l}{R/K - r_{\downarrow}}. \tag{4}$$

$$r_{\downarrow} = \frac{D_p}{D_p + 1} \cdot R/K. \tag{5}$$

Convergence of the Overlay

- From (8) and (9), we can see the larger the values of P(A) and $t_{\downarrow}$, the more likely the topology can self-converge, specifically:
  - P(A) is an increasing function with the increase of free upload bandwidth in the overlay;
  - Dp plays an important role on the extinction of unstable nodes, where is a partner selected as a temporary parent. This implies that those peers with larger substream degrees can also contribute to the convergence of the overlay;
  - Larger latency between peers help overlay convergence.

Convergence of the Overlay

- We simulated three different configurations to examine overlay convergence of Coolstreaming
- A contains 5% of 100 Mbps upload capacity peers; 20% of 10 Mbps peers; 25% of 512 Kbps peers; 50% of zero upload capacity peers
- B is 15%; 20%; 15%; 20%, respectively;
- C is 25%; 20%; 5%; 50%, respectively.
- The bit rate of the video stream is 512 Kbps
- There are six substreams (K=6)
- each peer has maximum ten partners (M=10)
- Buffer length is set to be 60 seconds
- cool-down period Ta is 10 seconds
- two thresholds of peer adaptation Ts and Tp to be fixed at 20 seconds

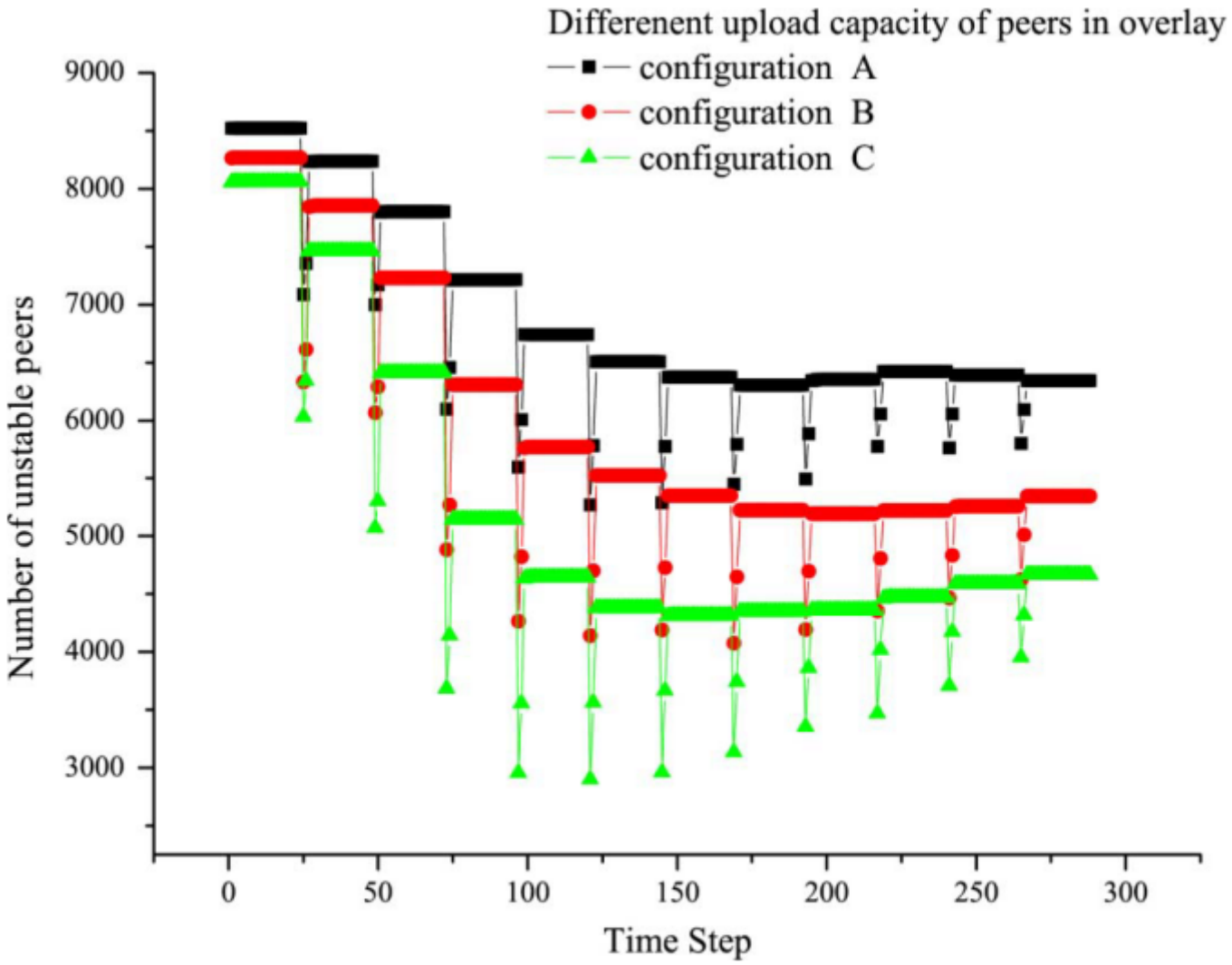


Fig. 9. Number of unstable peers along simulation time under different overlay configuration.

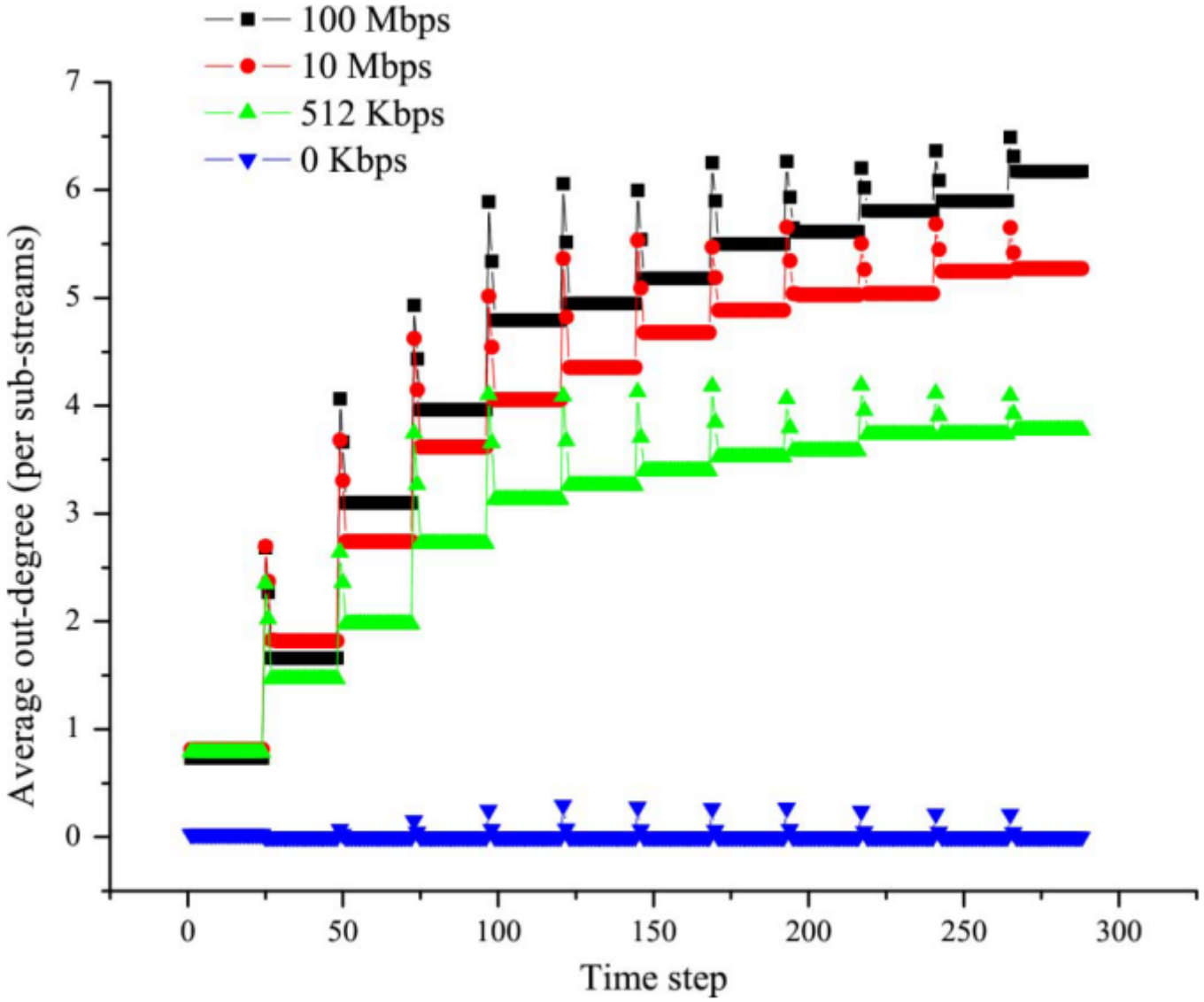


Fig. 10. Average out-degree for different type of peers along simulation time.

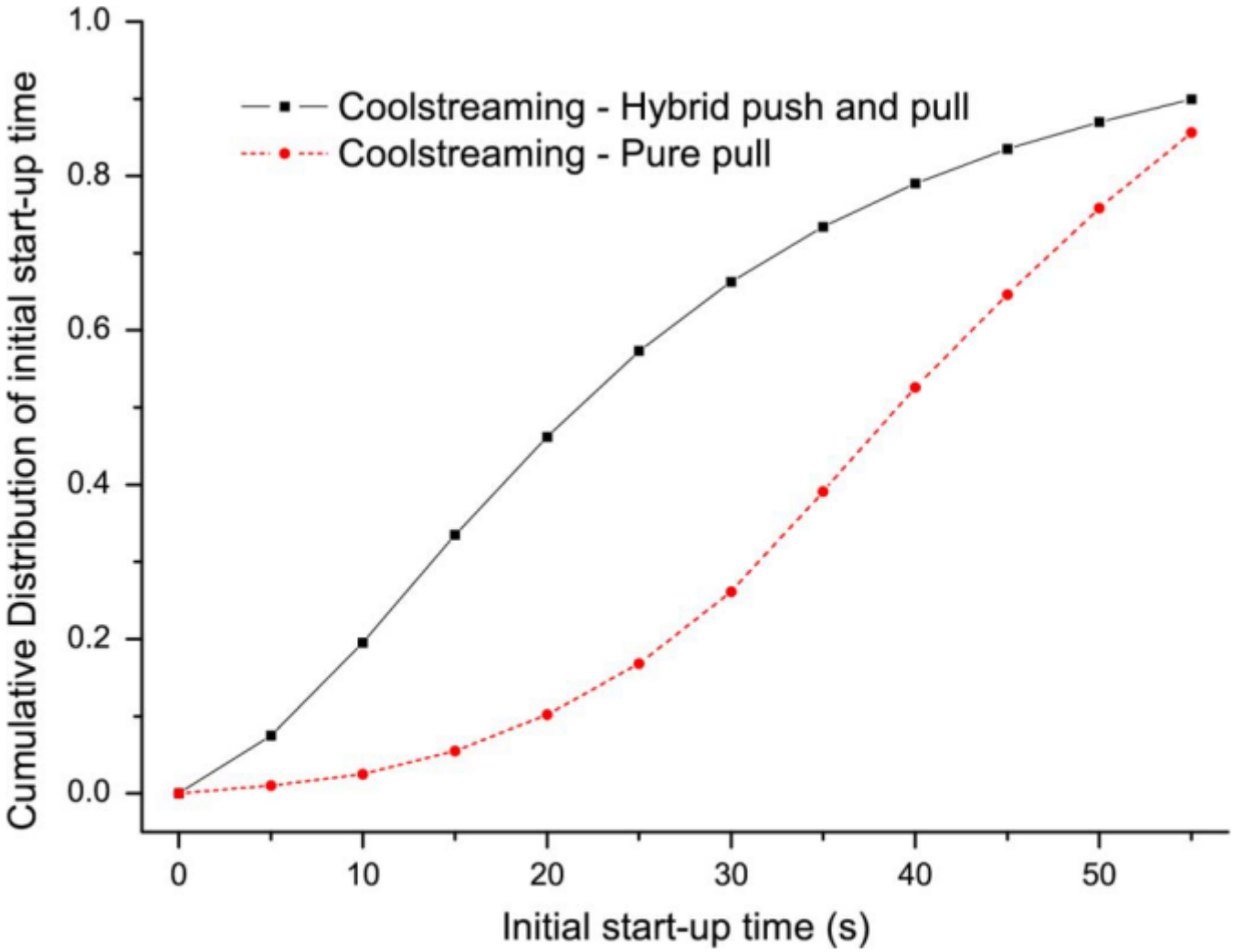


Fig. 11. Comparison of initial start-up delay between pure-pull based and hybrid push-pull.

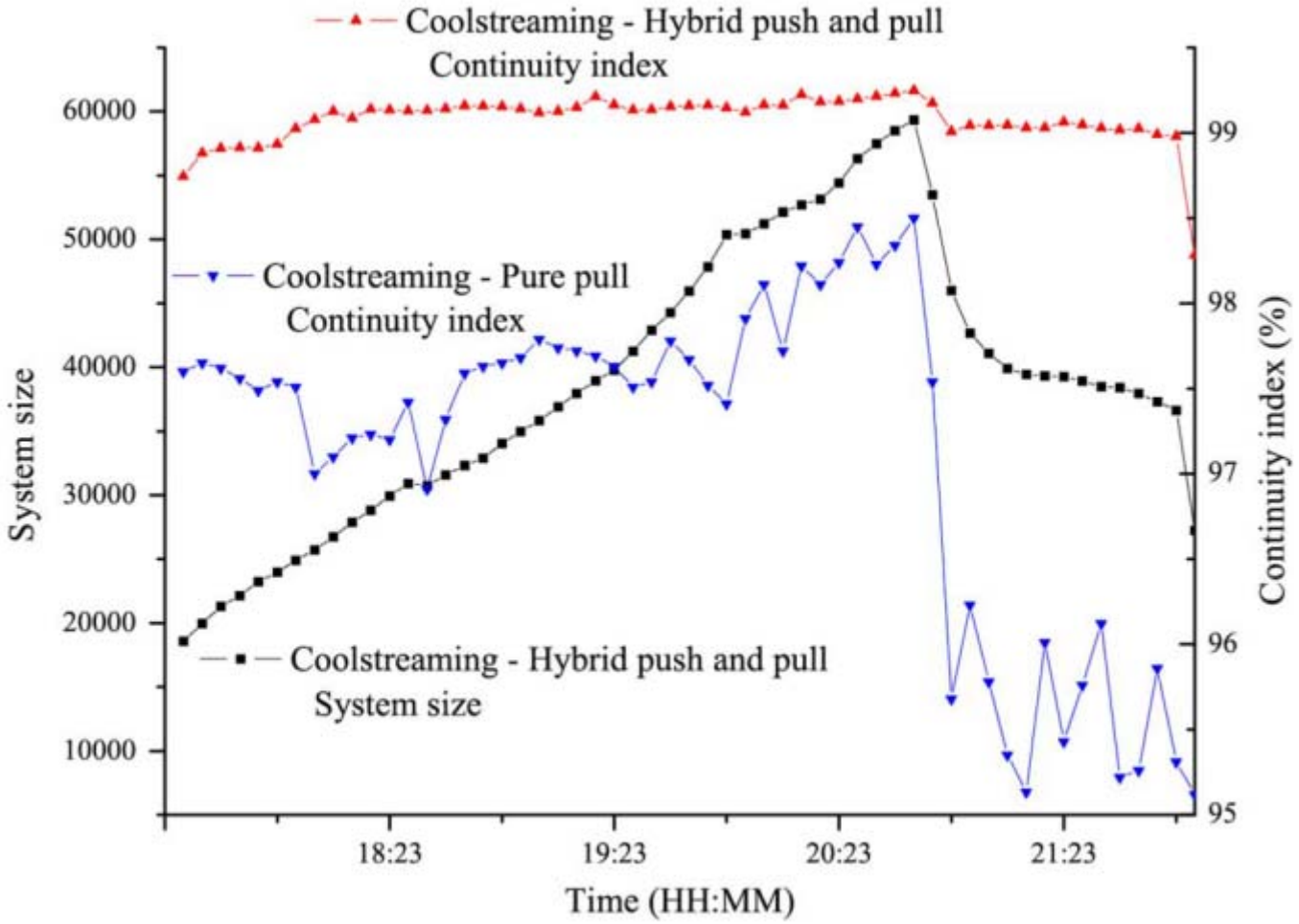


Fig. 12. Comparison of continuity index and number of users.

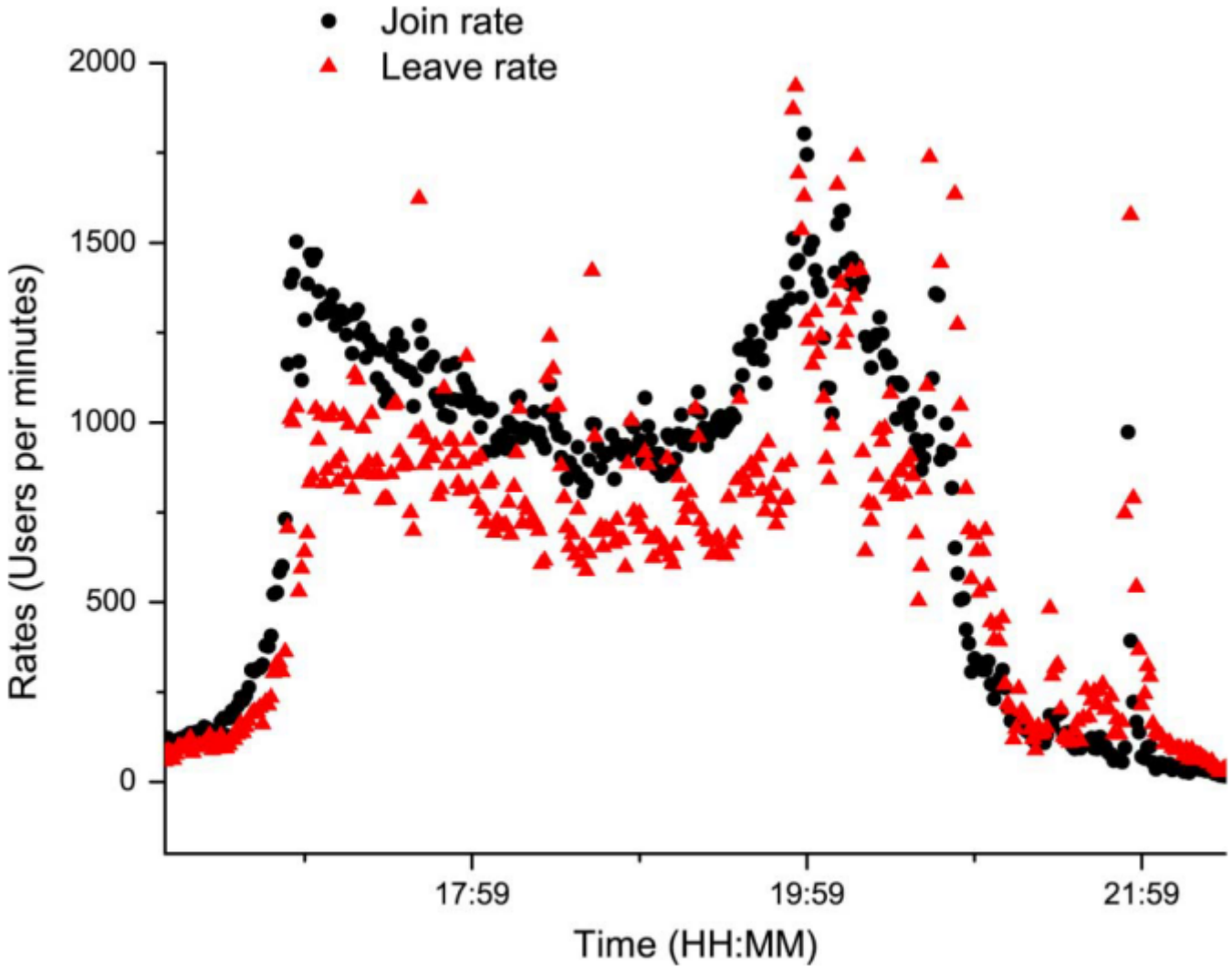


Fig. 13. Join rate, leave rate, and difference between the rates.

1. Introduction
2. Detail of Cool Streaming
3. System Dynamics
4. Results and Discussions
5. Conclusion

Conclusion

- Coolstreaming represents a milestone in P2P live video streaming systems
- It demonstrated practically that a simple random peer selection algorithm coupled with the multiple substream transmission technique has the potential to scale
- There are two points that we like to emphasize from this study
  - 1) a working system is essential in providing basic understanding
  - 2) there is a large number of practical problems that have to be dealt with in real engineering

- Q&A