Resource and Locality Awareness in an Incentive-based P2P Live Streaming System

Fabio Pianese and Diego Perino

France Telecom - Division R&D
Issy-les-Moulineaux, France

2nd P2P-TV SIGCOMM Workshop
August 31, 2007- Kyoto, Japan

IPTV: Streaming Live Media

P2P Live Streaming on the Internet

- Huge interest: reduce the cost of live media distribution
- High potential: self-scalability as users provide resources
- Easy deployment (compared to global native multicast)

Encouraging Results...

- Various *Tree-based* and *Mesh-based* approaches
- Efficient bandwidth utilization (w/ uniform upload capacity)

Several Issues are Still Open!

1. How to deal with widespread **upload heterogeneity**?
2. How to deal with **node transience** and **heavy churn**?
3. How to implement **awareness of network topology**?

Contribution

Upload Heterogeneity usually solved by **extrinsic** mechanisms (e.g. public bartering records, assume correct reporting, etc.)

$\Rightarrow$ *Not very effective in a real-world context!!!*

Our Goals

- Introduce **intrinsic bandwidth awareness**
- Introduce **intrinsic latency awareness**
- Have a system **intrinsically resilient to churn**

But... how to do that?

- Organize the peers in the system as a **dynamical mesh**
- Exploit the **side effects** of an **altruistic TFT incentive**

What?!? A TFT Incentive? Again? Please!

In live-streaming P2P services, all peers receive data at the same rate, e.g. that of the video encoded rate. Thus, discouraging resource rich peers from helping more.

– *Anonymous Reviewer #1*

[...] insisting on pure TFT in a streaming system does not give enough incentive to nodes with high upload bandwidth to contribute more. It is [...] unclear why authors insist on TFT.

– *Anonymous Reviewer #2*

[...] the mechanism tit-for-tat itself is very old.

– *Anonymous Reviewer #3*

Yes, an Altruistic TFT Incentive!

Pairwise incentives have been successfully employed in P2P content distribution applications, e.g. BitTorrent

Primary Purpose (but not for us!)

Enforce (some definition of) **fairness** in the resource utilization

Recently Discovered Side Effect

Nodes form **clusters** by amount of shared resources [1][2][3]

Hence, Our Idea!

Exploit this spontaneous clustering to improve the support of bandwidth heterogeneity in the node population (**PULSE**)

[1] Neglia, G. *et al.*, "A network formation game approach to study BitTorrent Tit-for-Tat", EuroFGI 2007
[2] Gai, A. *et al.*, "Stratification in P2P Networks: Application to BitTorrent", ICDCS 2007
[3] Legout, A. *et al.*, "Clustering and Sharing Incentives in BitTorrent Systems", SIGMETRICS 2007

So, How To Encourage Altruism?

Insight: Randomness Is Key!

1. Natural churn: peers come and go all the time
2. Induced churn: connections are renegotiated often
3. Timeliness: the playout deadline is not far away
4. Retaliation: if you never give, you will rarely receive

Altruism spread among many peers discourages leeching

- Not enough data per peer to sustain continuous streaming

Lack of altruism may result in random temporary starvation

- *Accidents a-happen, y'a-understand? <evil grin>*

Altruism is a way to strengthen the ties between nodes

$\Rightarrow$ so: must choose the targets for altruism appropriately

Optimistic Tit-for-Tat in BitTorrent

We set out to adapt the BT TFT incentive to our new context

- Renegotiate all connections every EPOCH = 30s
- Number of neighbor *slots* is fixed (e.g four) [impl. specific]
- Serve the best peers in terms of contribution (last EPOCH)
- Slot reserved for *random optimistic* selection

Changes are required to:

- Improve robustness of data reception (avoid starvation!)
- Promote synchronization of *rich* nodes on recent data
- Avoid the under-utilization of available upload capacity

Optimistic Tit-for-Tat in PULSE

PULSE implements a modified TFT incentive:

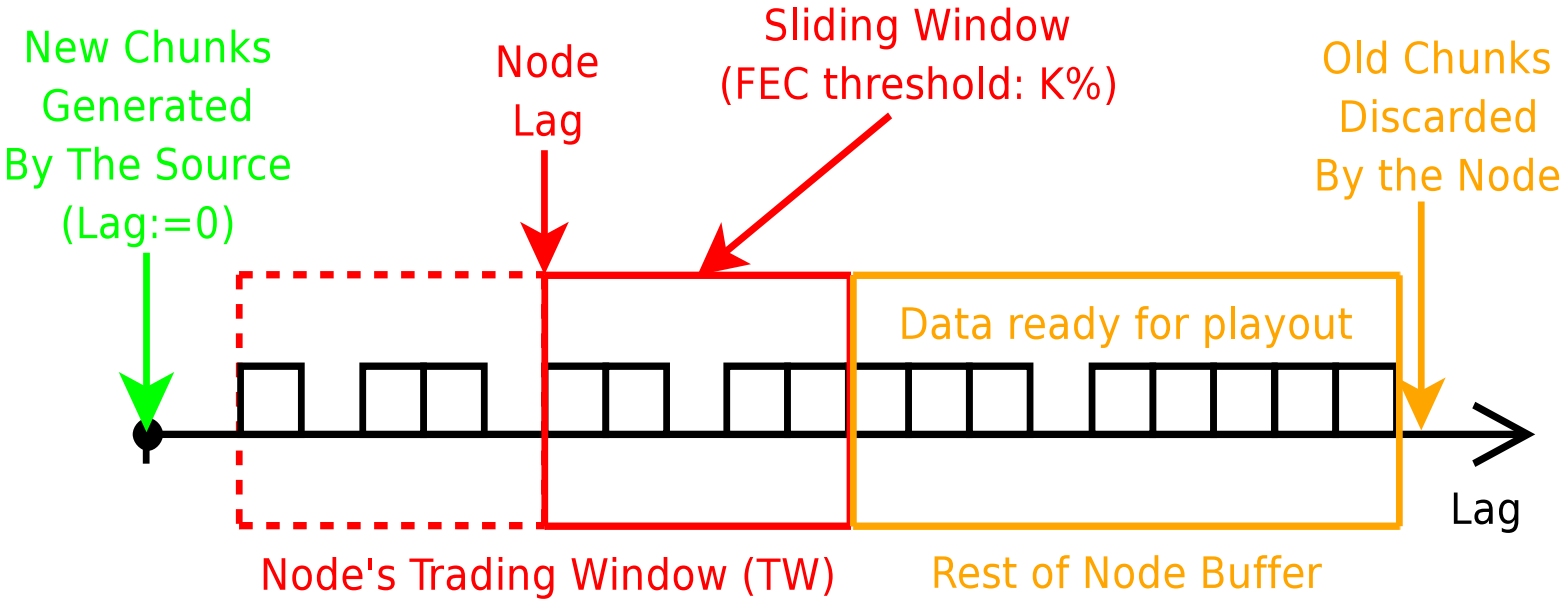
- Renegotiate all connections every EPOCH = **2-5s**
- Number of TFT neighbor *slots* is fixed (four)
- Serve the best peers in terms of contribution (last EPOCH)
- Slot reserved for optimistic selection **w/ shared interest**
  - Choice is performed based on the **buffer data range**
- **As upload permits**, serve peers that need **older data**
  - Additional altruism, again based on **buffer data range**

Also: the buffer implements a **sliding window** mechanism

- Deliver a continuous sequence of chunks to media player

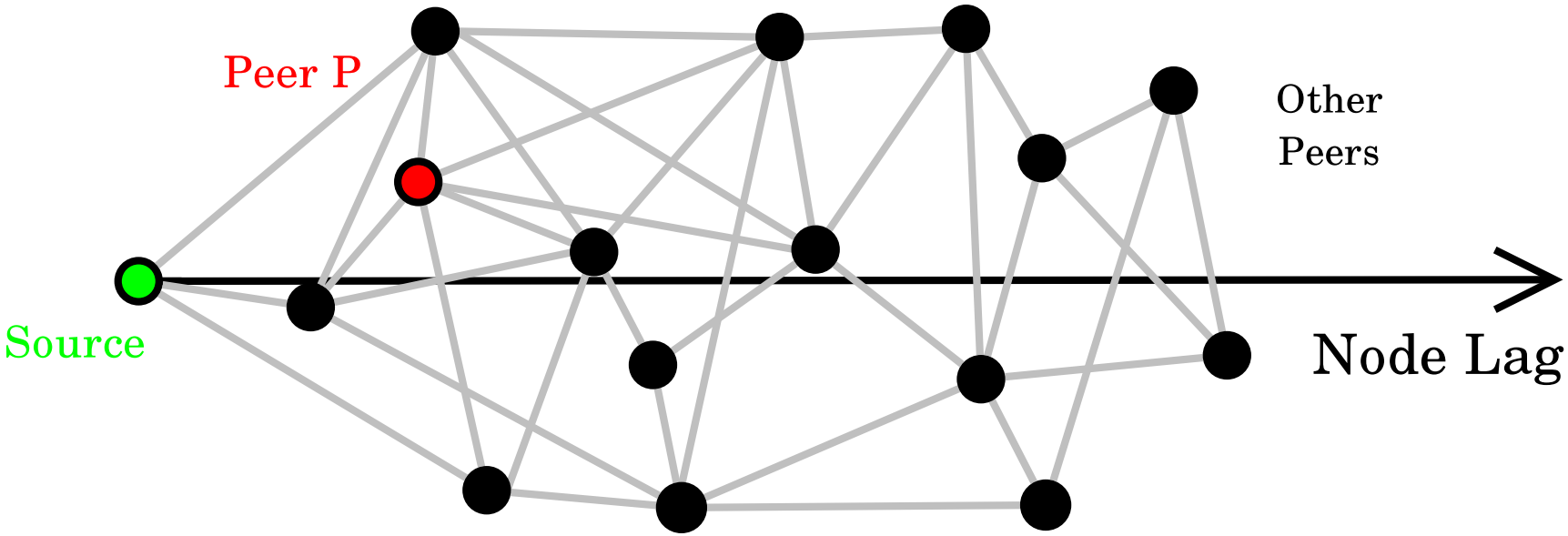
About the Structure of the PULSE Node Buffer

The stream is composed by a continuous sequence of chunks
Classic FEC is applied by the source: $K\%$ of chunks are parity

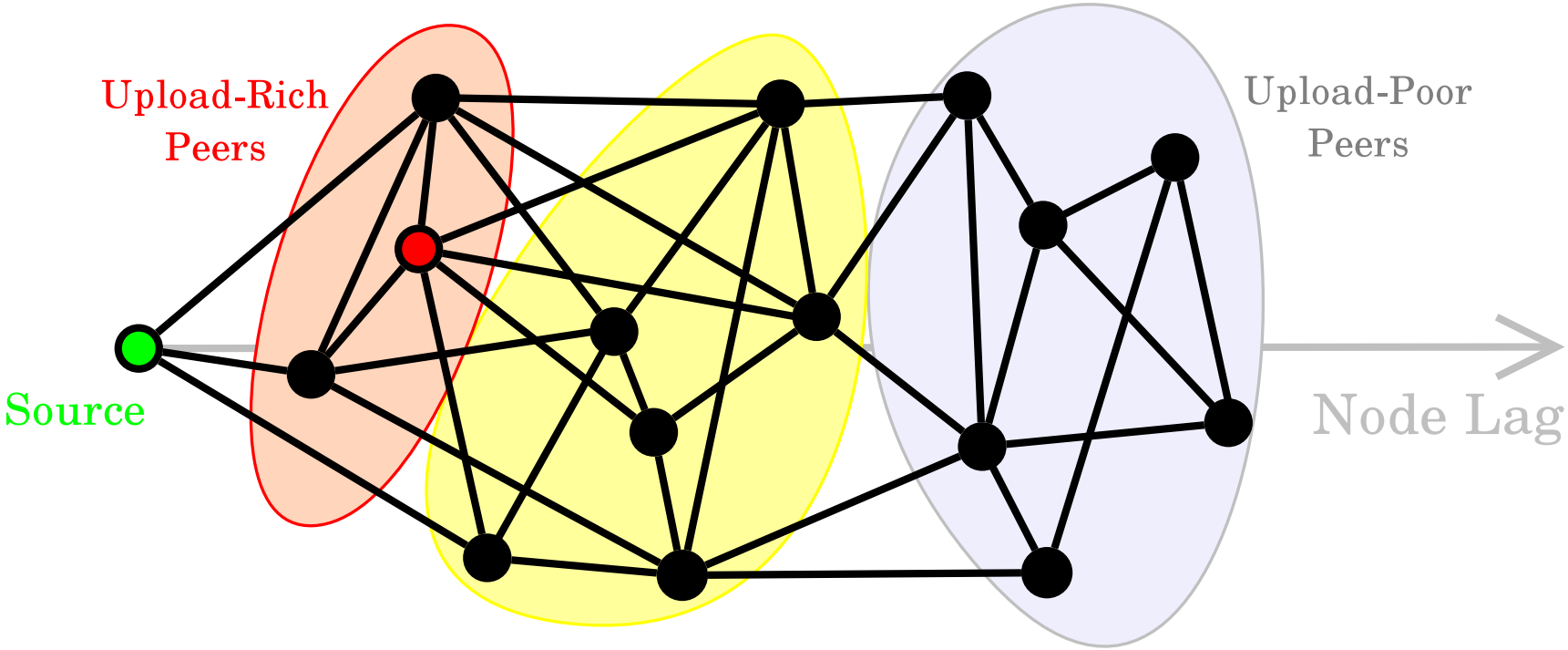


The buffer implements a sliding window mechanism: it may slide to the left only when *less than* $K\%$ chunks are missing

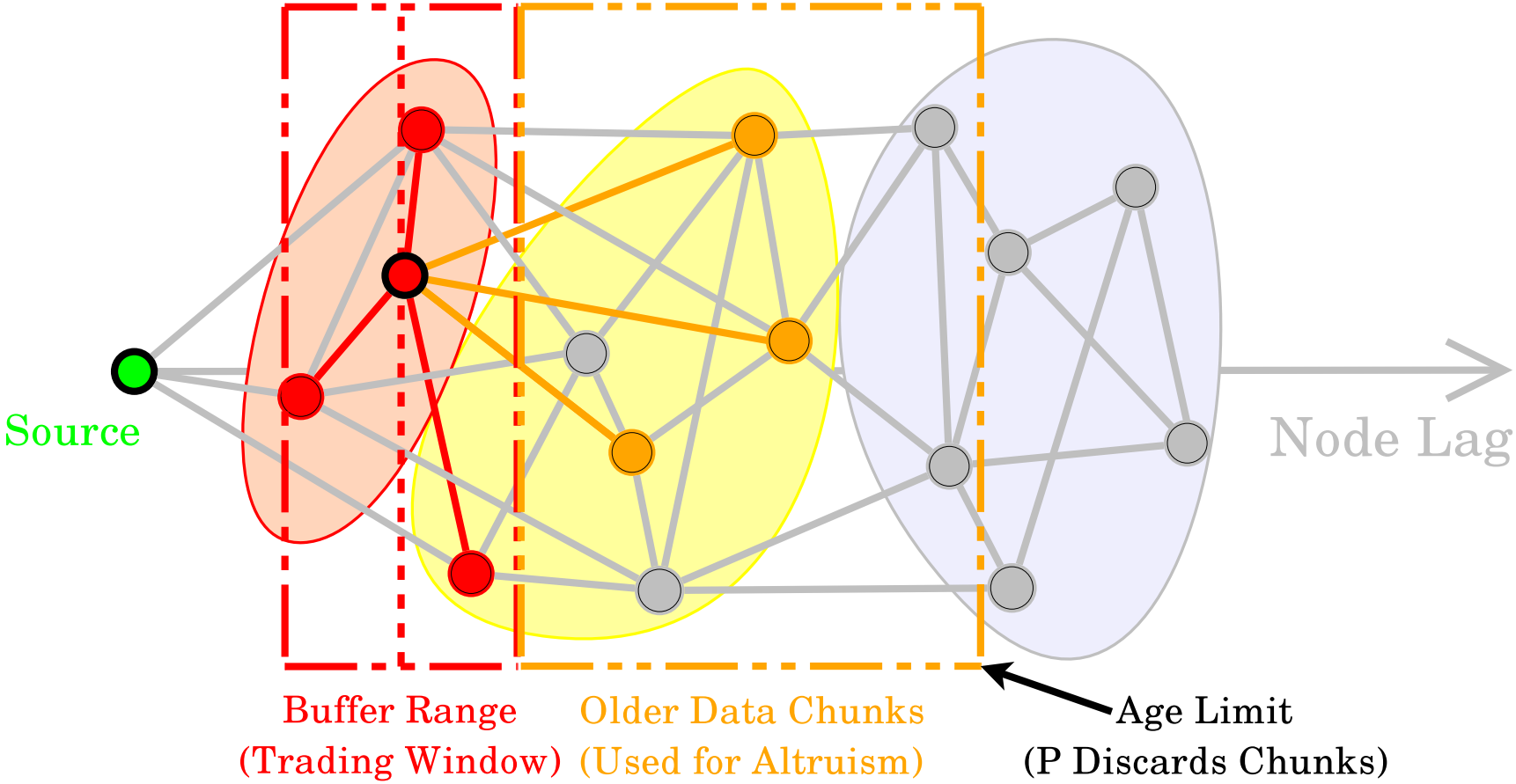
Mesh + ...



Mesh + Clustering + ...



Mesh + Clustering + Targeted Altruism = ...



... = Bandwidth Awareness!

Adaptive lag-based clustering is induced by the TFT incentive

- Nodes layered by decreasing shared upload bandwidth

Optional altruism improves the global performance

- Resourceful nodes help nodes which are nearly as rich
- TFT retribution is enhanced for the altruistic nodes!

Expected results:

- Wide and short chunk distribution trees (on average)
- The resourceful nodes obtain a lower and stable lag
- Near-optimal bandwidth allocation even under scarcity

... = Locality Awareness!

Obtained by introducing a *latency bias* parameter C

- We measure latency of neighbor nodes (control msgs)
- It's very easy to influence the optimistic TFT selection
- Idea: add preference for topologically near neighbors!

Expected results:

- Average improvement in the topological awareness
- Clustering biased by both bandwidth and latency

The PULSE Prototype Node

Coded in python, available on http://pulse.netofpeers.net/

- Uses the Twisted framework to manage the sockets
- Data exchanges use TCP, control messages on UDP

Code is licensed under GNU LGPL, you are free to play with it

- Early snapshot, not very polished, use at own risk...
- Improvements made to it are *very much* welcome!

Testbed Setup

Emulab: Grid'5000 Testbed (G5K)

- Up to 800 nodes on up to 400 dedicated machines
- Almost-flat network topology, quite uniform latency
- Bandwidth limitation performed at application level

Internet: Planet-Lab Testbed (PL)

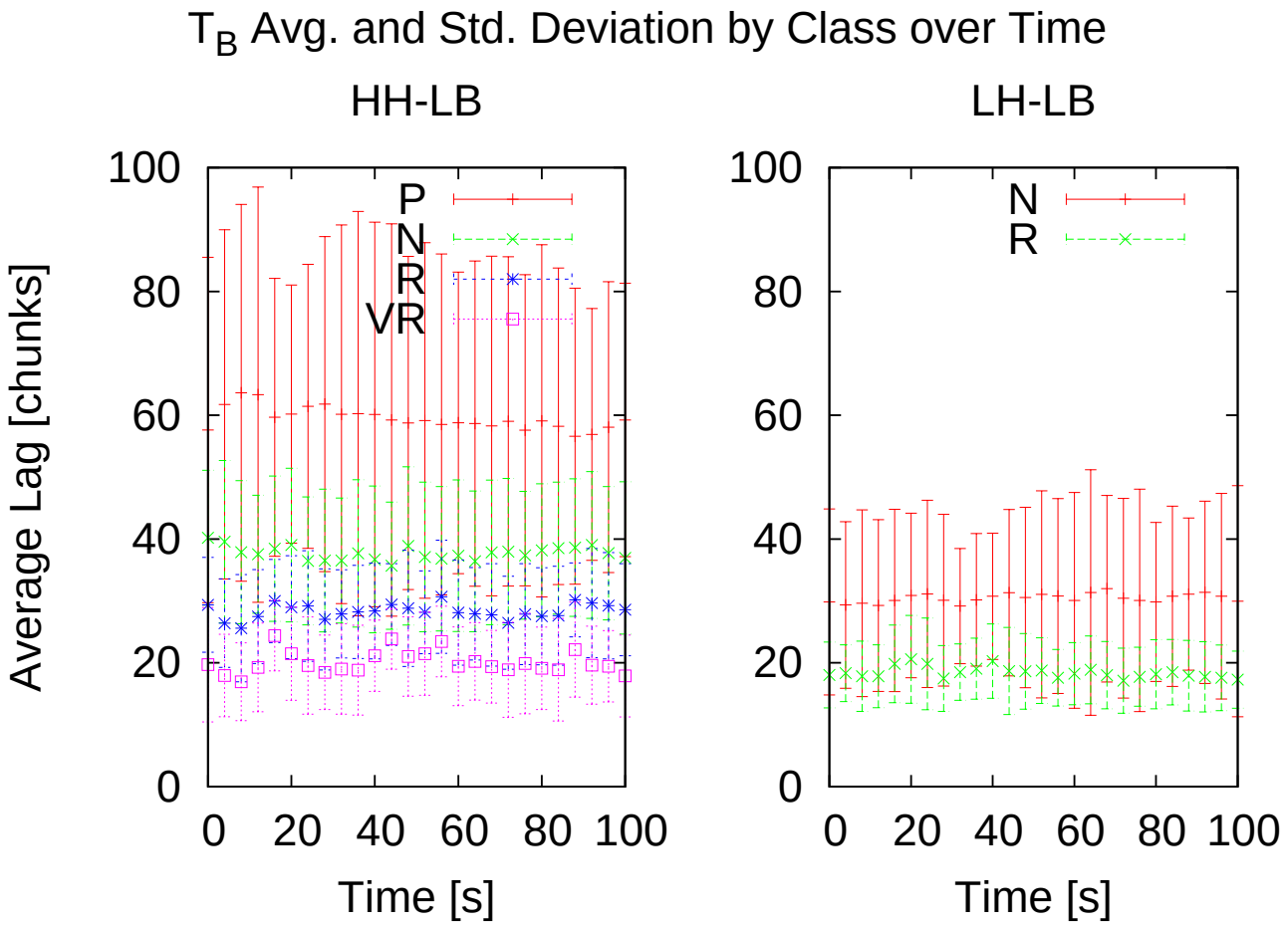
- Up to 200 nodes on machines all around the world
- Latencies assumed as typical of a real-world scenario
- No bandwidth limitation performed by the application
  - CPU of our PL slice proved to be the actual bottleneck
  - CPU shortage results in artificial limitation of BW use

Parameters

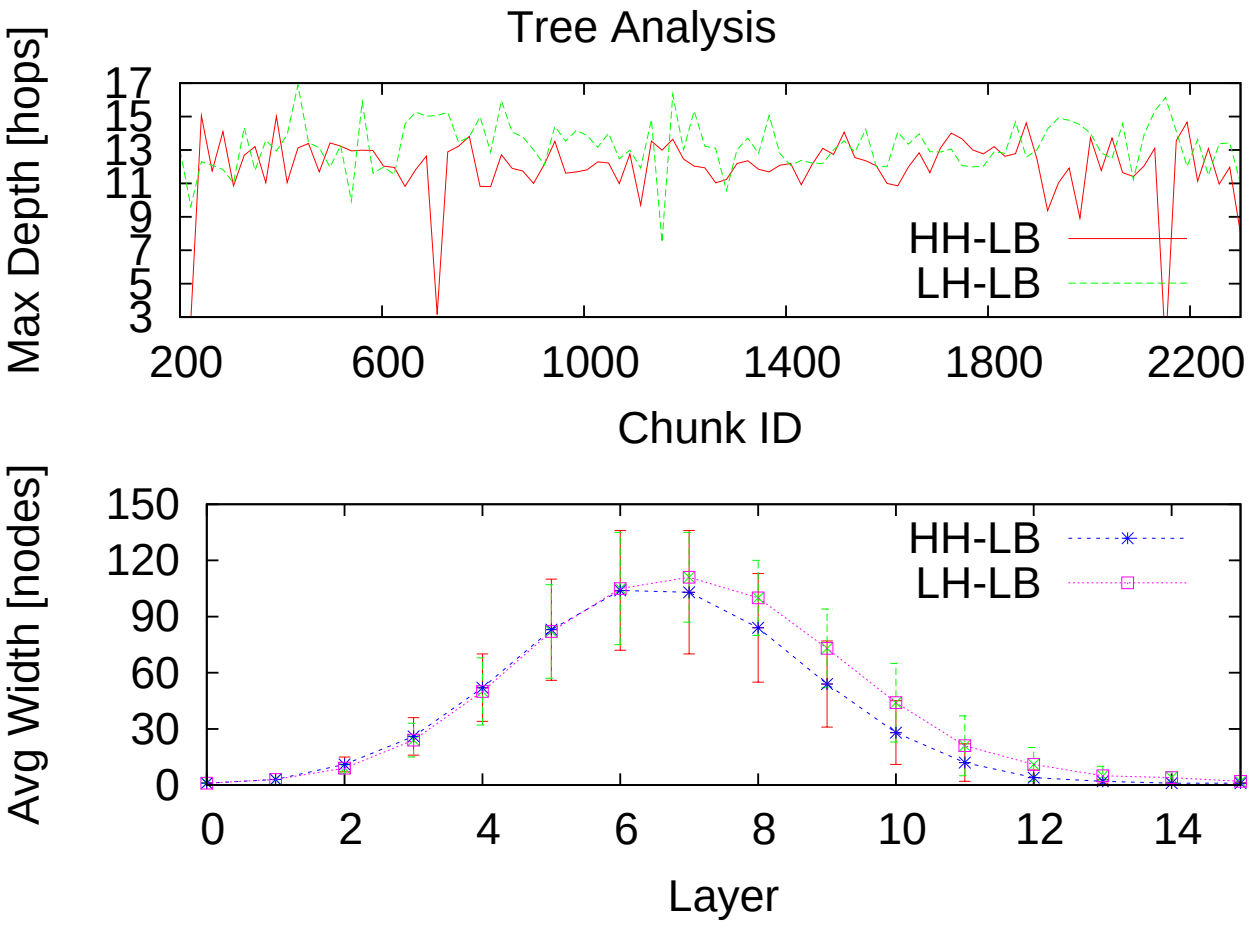
1. Chunk Rate: 16 chunks/s (G5K) - 4 chunks/s (PL)
   - Stream Rate (*SBR*) 256 Kbps
2. Outbound Connections: 12 total ($N_A = 8$, $N_{TFT} = 4$)
3. Grid'5000 BW distribution chosen among the following:
   - HH-LB - 4% (VERY RICH) 10*SBR, 20% (RICH) 3*SBR, 21% (NORMAL) SBR-2*SBR, 55% (POOR) $\frac{SBR}{2}$-2*SBR
     Average Node Capacity in this Scenario: 1.04*SBR
   - LH-LB - 20% (RICH) 2*SBR-SBR, 80% (NORMAL) SBR-2*SBR
     Average Node Capacity in this Scenario: 1.2*SBR

   Source always has bandwidth 4*SBR (UL-DL)
4. $TW = 32$ chunks; FEC rate=20%; EPOCH=2s
5. Requests: Max outstanding/peer=2 - Timeout=0.5s

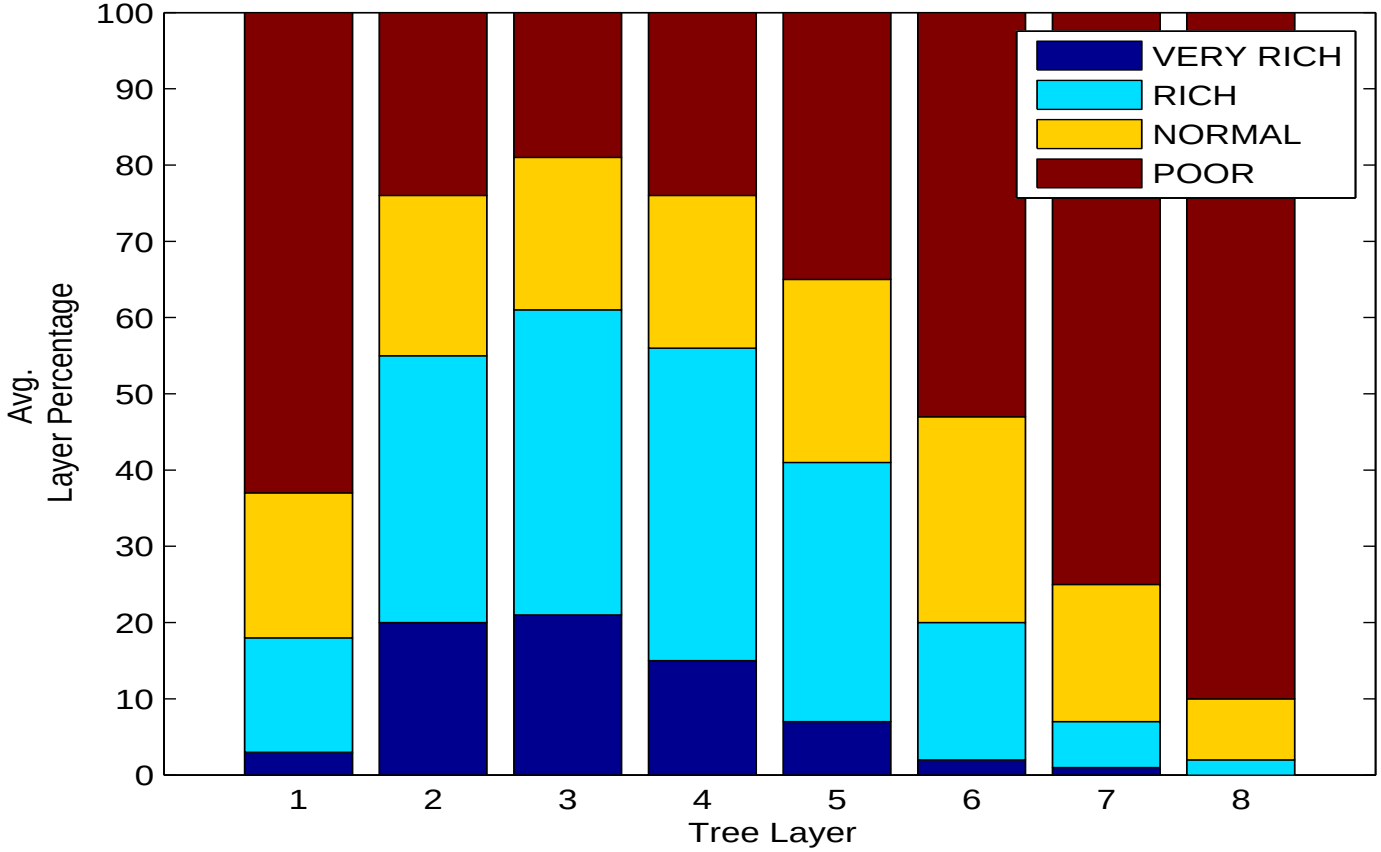
Evaluating Bandwidth Awareness (G5K)



Chunk Distribution Trees (G5K) - 800 Nodes



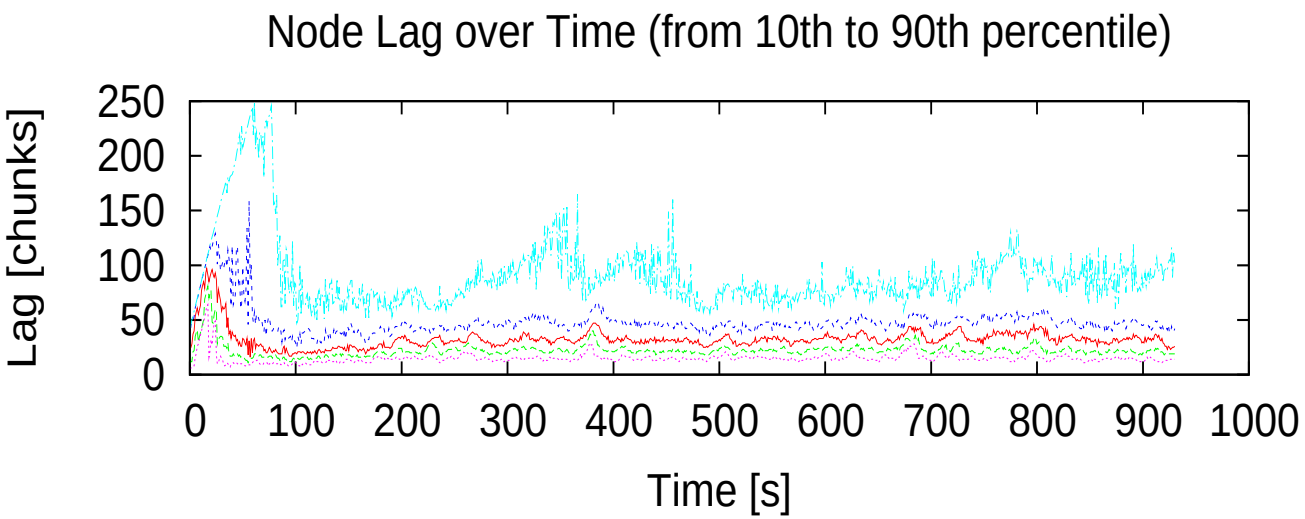
HH-LB: Tree Layer Composition (by BW Class)

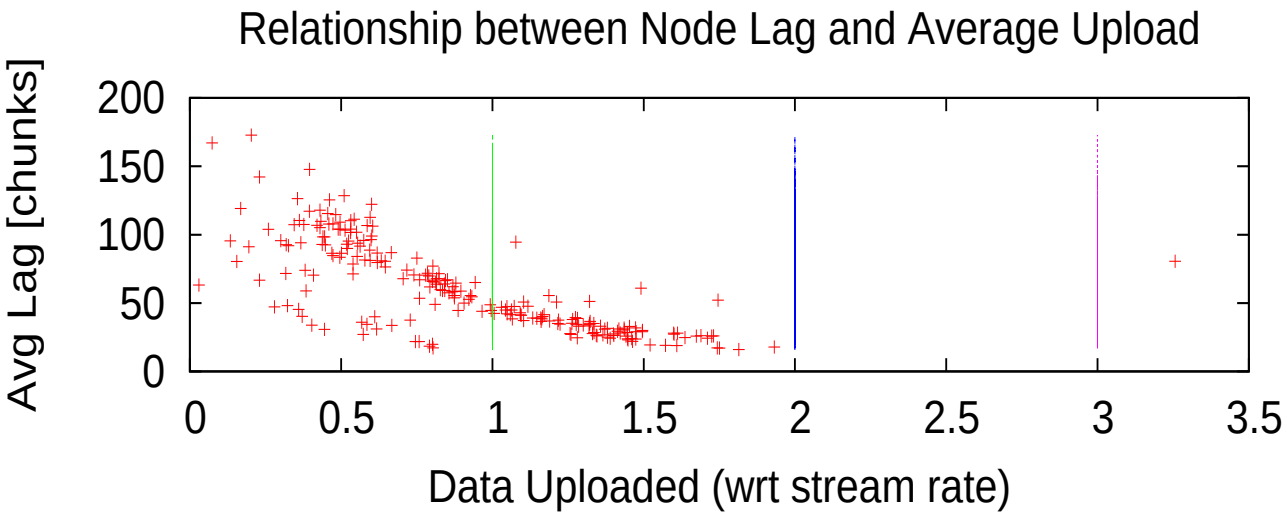


Peer Selection at source is not TFT-based (random, lag range)

- Layer 1 is not “optimized”, but the following are

Bandwidth Awareness on PL - 200 Nodes





Introducing the Latency Bias (PL)

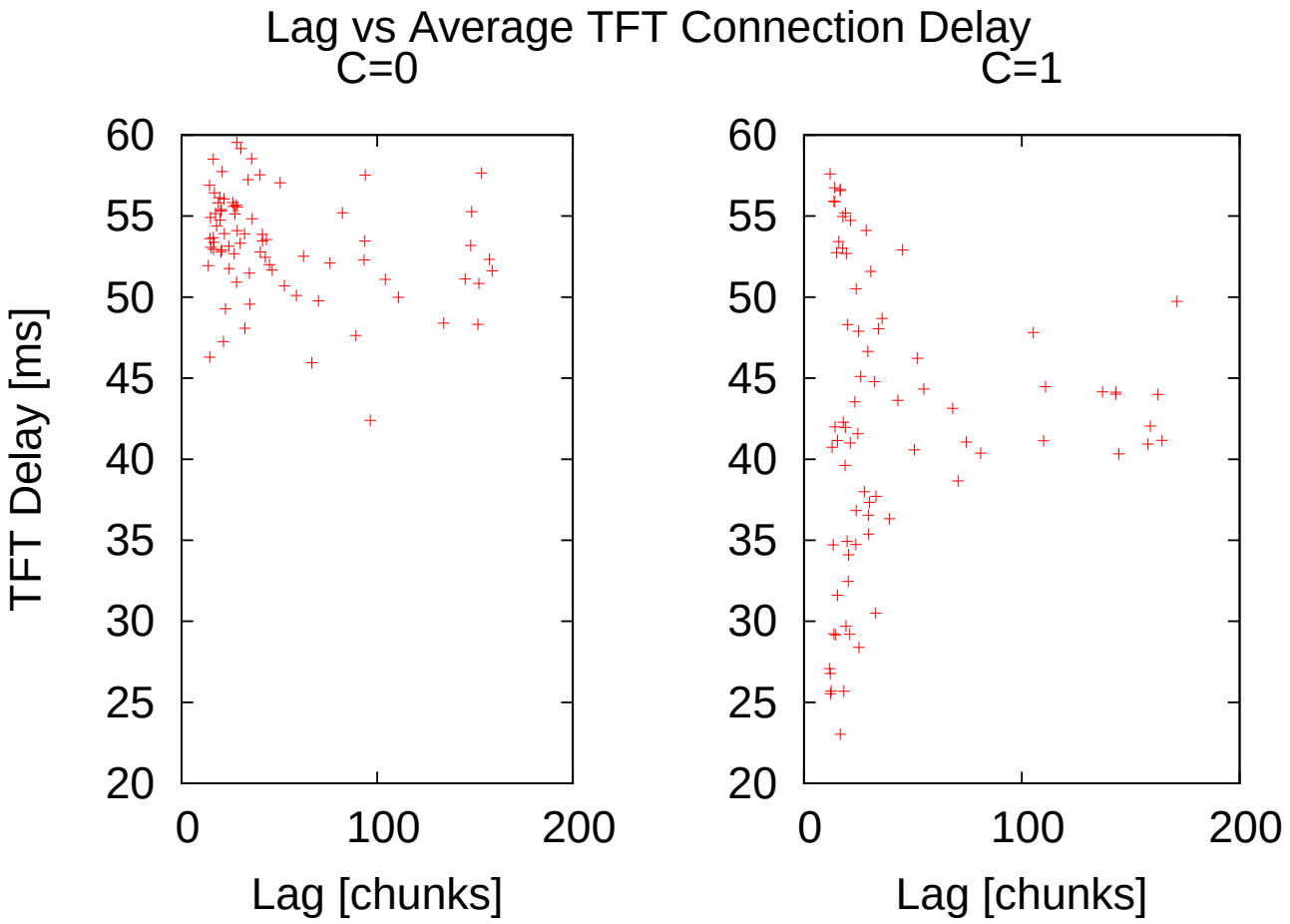
We experimented with several values, $C = 1$ is presented

Lag Percentile	10%	30%	50%	70%	90%
C=0	12.31	18.10	26.18	37.70	61.08
C=1	10.53	14.47	18.89	27.22	49.39

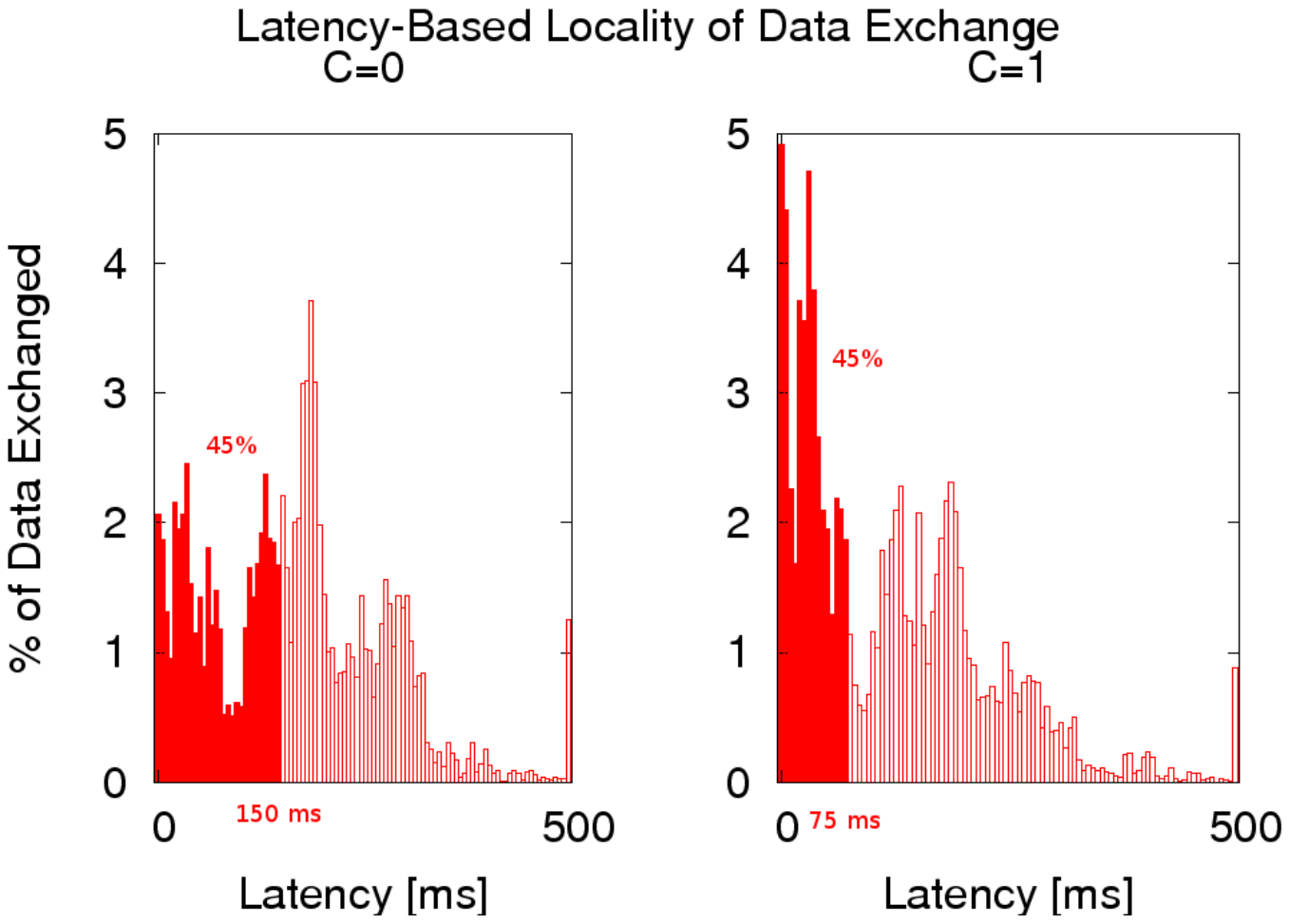
Table shows average x^{th} percentile lag (in chunks)

- Introducing latency awareness also benefits overall lag!
  - "shorter" connections $\Rightarrow$ faster *control* exchanges
  - faster control exchanges $\Rightarrow$ higher upload utilization
- Percentile lag gain ranges from 16% (10th) to 38% (70th)

Latency Awareness of TFT Connections (PL)



Latency Awareness of Data Transfers (PL)



Claims

The system showed encouraging results for different BW scenarios, arrival patterns and population sizes

- TFT is effective to discriminate nodes based on upload
- Altruism + lag feedback help consolidate clustering
- Mesh of connections supports very high churn rates

Results:

- PULSE can implicitly support BW heterogeneity
  - Incentives can be used as a metric for optimization!
- Introducing locality optimization is easy and effective

Questions?

Incentives: Definitions

We will indicate the buffer lag of peer x as $T_B(x)$
The width of the Trading Window is defined as TW

Data Buffer: Distance and Interest Overlap

Buffer Distance between peers a and b:
$D(a, b) = T_B(a) - T_B(b)$
Buffer Interest Overlap:

$$I(a, b) = \begin{cases} TW - |D(a, b)| & \text{if } |D(a, b)| \leq TW \\ 0 & \text{otherwise} \end{cases}$$

If $D(a, b) > TW$ we say that a is **ahead of** b, b is **behind** a
$\Rightarrow$ a can serve b (*older* chunks), b cannot serve a

Pairwise Latency (measured by control messages)

Network latency between peers a and b: $L(a, b)$

PULSE: Pairwise Incentive

Optimistic TFT for Live Streaming: N_{TFT} slots

The best $N_{TFT} - 1$ contributors of last EPOCH will be chosen

Optimistic selection slot at peer a:

- Among all nodes b for which $I(a, b) > 0$ choose the one for which $BI = I(a, b) - C \cdot L(a, b)$ is highest

BI is the **biased interest** function. C is the **latency bias**

Requests by these N_{TFT} peers are served with high priority

PULSE: Pairwise Incentive

Contribution-Aware Optional Altruism: N_A slots

Keep a cumulative score H about all the known nodes

- Increase score when unexpected chunk received
- Decrease score when issuing altruistic contribution

Altruistic selection at peer a:

- Among all nodes b that are behind a
  choose the F nodes with the highest H score

Requests by these N_A peers are served with low priority

Related Work

Awareness to Upload Heterogeneity

Rely on autonomous/external contribution policing

- Multiple-ESM [4]: nodes are **entitled** to join trees based on the upload they contribute (self-enforced)
  - Nodes can "cheat" and fail to behave properly
  - Changes in tree topology cause disruption
- Bartering schemes - eg. [5] - (externally enforced)
  - Not evaluated in practice, to our knowledge

Periodically replace the least-performing neighbors

- Coolstreaming/DONet [6] iteratively updates its mesh

[4] Sung, Y. *et al.*, "Enabling contribution awareness in an overlay broadcasting system", SIGCOMM 2006
[5] Pouwelse, J. *et al.*, "Real-time Video Delivery using Peer-to-Peer Bartering Networks and Multiple Description Coding", SMC 2004
[6] Zhang, X. *et al.*, "CoolStreaming/DONet: A Data-driven Overlay Network for Peer-to-Peer Live Media Streaming", INFOCOM 2005

Related Work

Resilience to Churn

Substantial consensus: meshes behave better under churn

Awareness to Underlying Topology

Information about topology is hard to gather at application layer!

- Relying on inline latency measurements (ping, etc.)
  - Low accuracy, but simple to implement
- Relying on external mechanisms (landmarks, etc.)
  - Rich literature on calculating Internet distances
  - Better accuracy, but requires an external subsystem

Both methods are easily adapted to mesh-based systems [7]

[7] Chang, B. *et al.*, "Refine DONet's Overlay with Network Distance Estimation", WRAIPS 2006