Resilient Peer-to-Peer Streaming

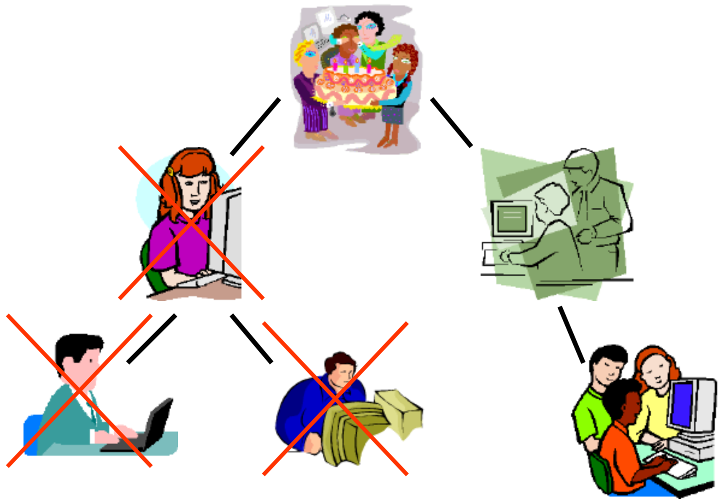
Venkata N. Padmanabhan
Helen J. Wang
Philip A. Chou

CoopNet Project
Microsoft Research
ICNP 2003

Problem Statement

- Problem:
  - live Streaming to a large audience in the wide-area Internet
- Peer-to-Peer Multicast is appealing
  - Self scaling
  - Easy to deploy, low cost
- A key challenge in supporting live streaming: *resilience*
  - Unreliable peers

Existing Peer-to-Peer Multicast

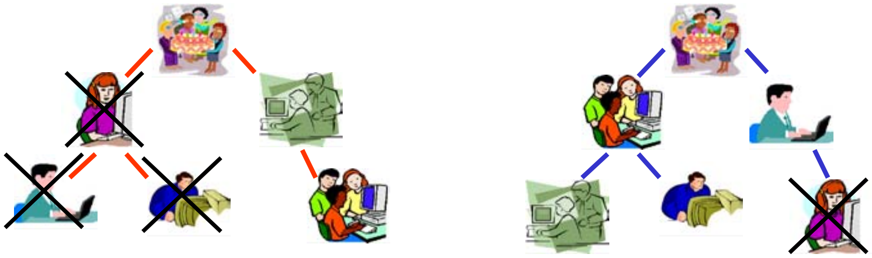


Vulnerable to node departures and failures

Our Approach

- Resilience through redundancy
  - Redundancy in network paths
  - Redundancy in data
- Place minimal demands on the peers
  - Only involve interested peers for traffic forwarding
  - Peer contributes only as much upstream bandwidth as it consumes downstream
    - natural incentive structure (enforcement is a hard problem!)

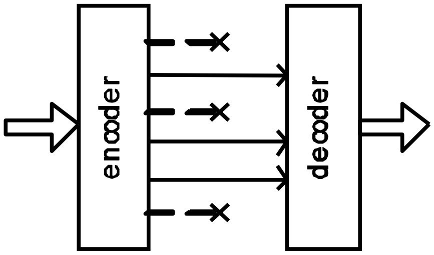
Path Redundancy: Multiple, Diverse Distribution Trees



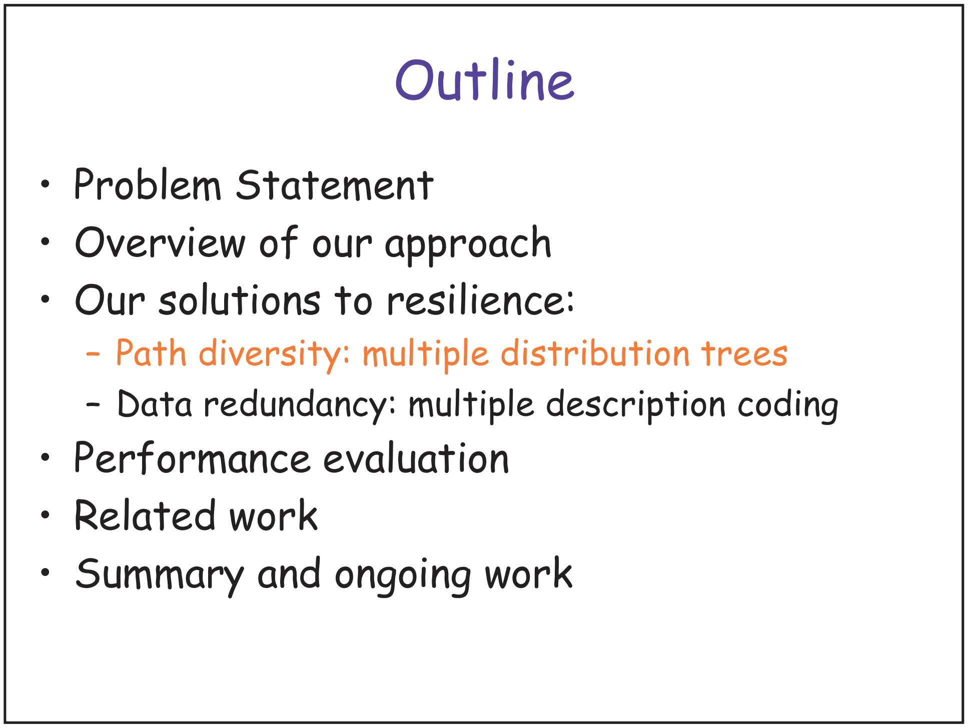
Tree diversity provides robustness to node failures.

Data Redundancy: Multiple Description Coding

MDC



- MDC codes data stream into *independent* descriptions (or substreams), *any* subset of which is decodable
- Intelligent striping + FEC

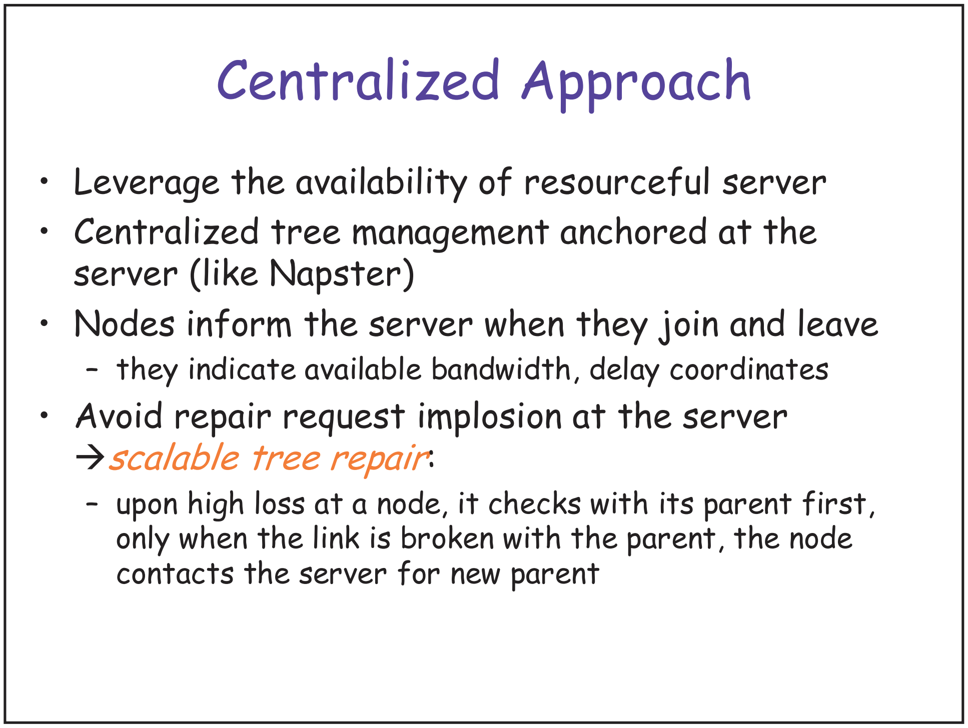


Outline

- Problem Statement
- Overview of our approach
- Our solutions to resilience:
  - Path diversity: multiple distribution trees
  - Data redundancy: multiple description coding
- Performance evaluation
- Related work
- Summary and ongoing work

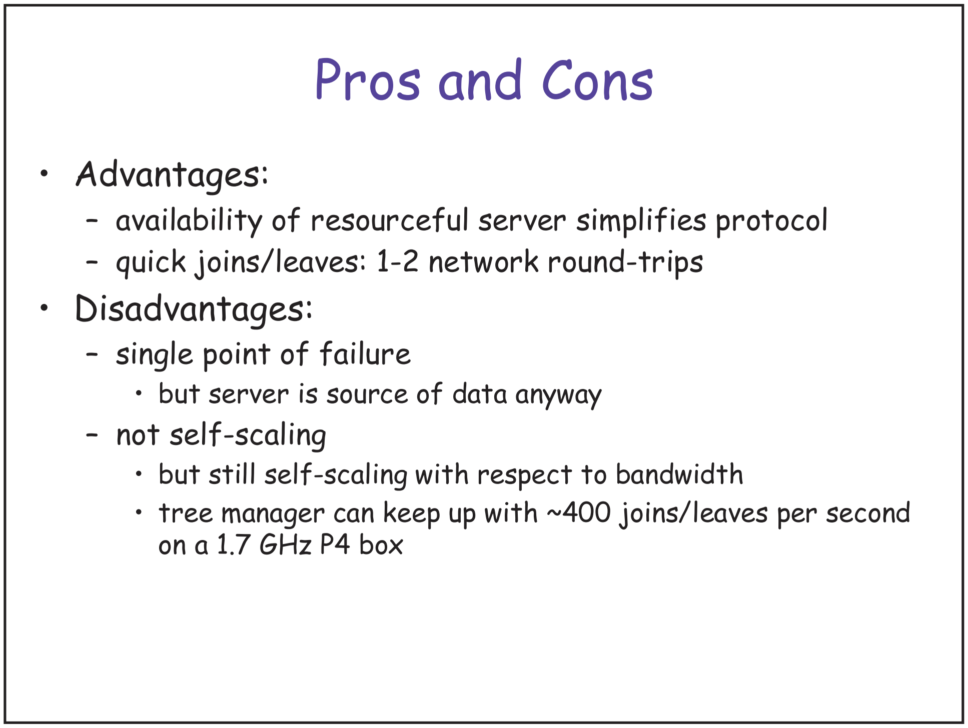
Tree Management: Goals

- Short trees
  - Fewer ancestors →less disruption
- Diversity
- Key to robustness
- Efficiency (stretch factor and link stress)
- Quick join and leave
- Scalability
- Conflicts:
  - Shortness vs. Efficiency
  - Diversity vs. Efficiency
  - Speed vs. Scalability



Centralized Approach

- Leverage the availability of resourceful server
- Centralized tree management anchored at the server (like Napster)
- Nodes inform the server when they join and leave
  - they indicate available bandwidth, delay coordinates
- Avoid repair request implosion at the server →*scalable tree repair*:
  - upon high loss at a node, it checks with its parent first, only when the link is broken with the parent, the node contacts the server for new parent



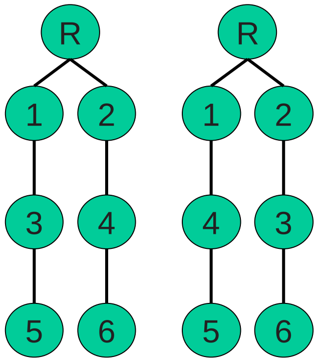
Pros and Cons

- Advantages:
  - availability of resourceful server simplifies protocol
  - quick joins/leaves: 1-2 network round-trips
- Disadvantages:
  - single point of failure
    - but server is source of data anyway
  - not self-scaling
    - but still self-scaling with respect to bandwidth
    - tree manager can keep up with ~400 joins/leaves per second on a 1.7 GHz P4 box

Randomized Tree Construction

- For shortness: the server places a joiner as high up in the tree as possible
- For diversity: randomly selects eligible parents who still have bandwidth to take more children
- Reported in our NOSSDAV '02 paper

Why is this Suboptimal?

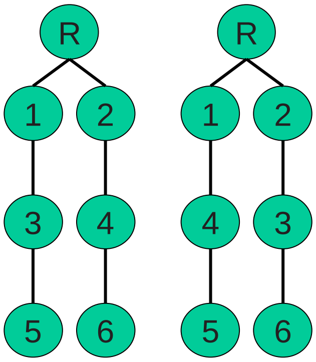


- We ask nodes to contribute only as much bandwidth as they consume
- So T trees ⇒ each node can support T children in total
- Q: how should a node's out-degree be distributed?
- Randomized tree construction tends to distribute the out-degree randomly
- This results in deep trees that not very bushy

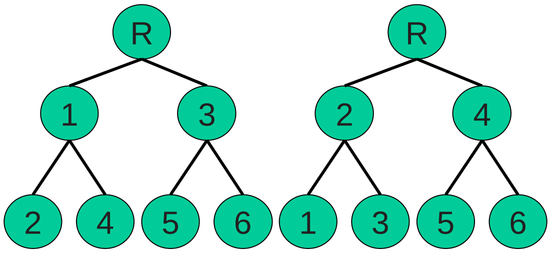
Deterministic Tree Construction

- Motivated by SplitStream work [Castro '03]
  - Lots of leaves -- cumulative upstream bandwidth abundant
  - A node needs be an interior node in just one tree;
  - Diversity can be achieved by making the interior nodes of the trees disjoint
- Distinctions:
  - Deterministic tree construction guarantees the disjointness, i.e., diversity, of the set of interior nodes across the trees
  - Only interested nodes are involved in forwarding

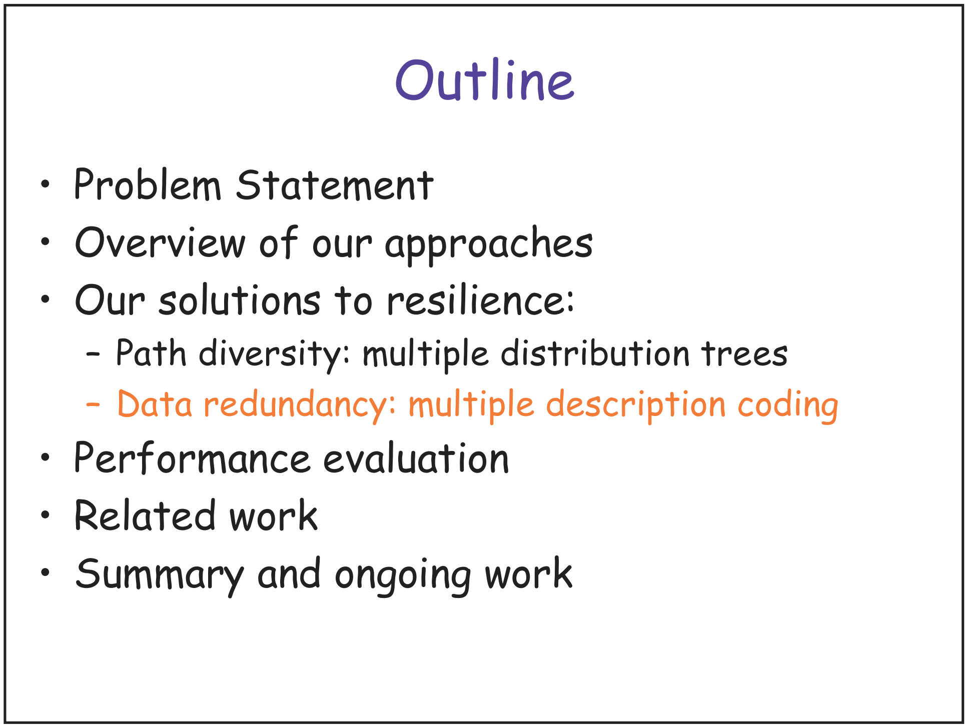
Randomized vs. Deterministic Construction



(a) Randomized construction

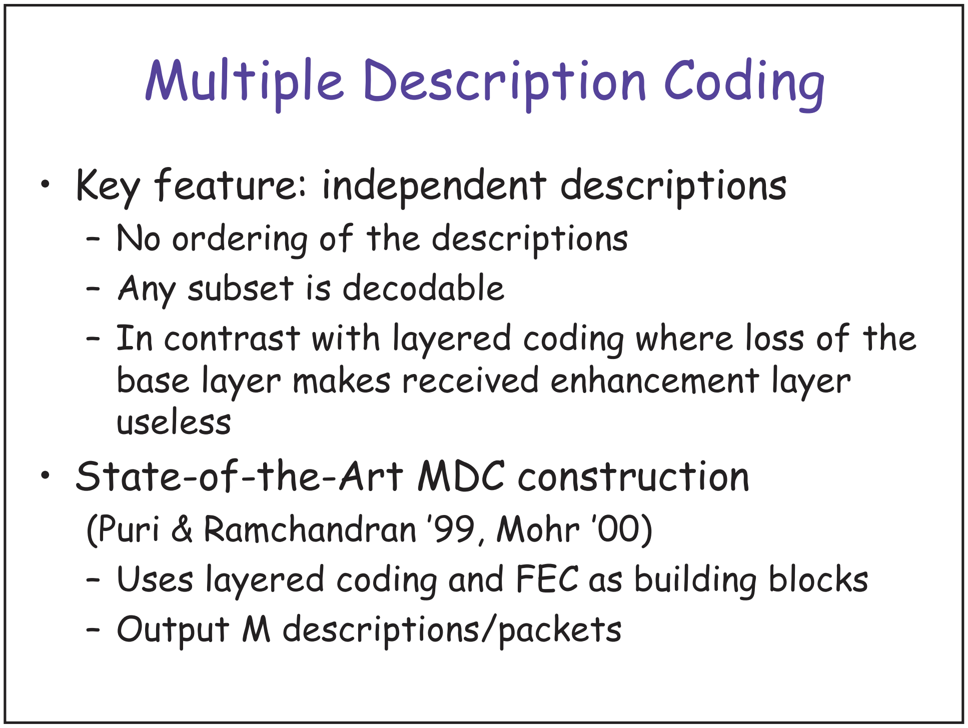


(b) Deterministic construction



Outline

- Problem Statement
- Overview of our approaches
- Our solutions to resilience:
  - Path diversity: multiple distribution trees
  - Data redundancy: multiple description coding
- Performance evaluation
- Related work
- Summary and ongoing work



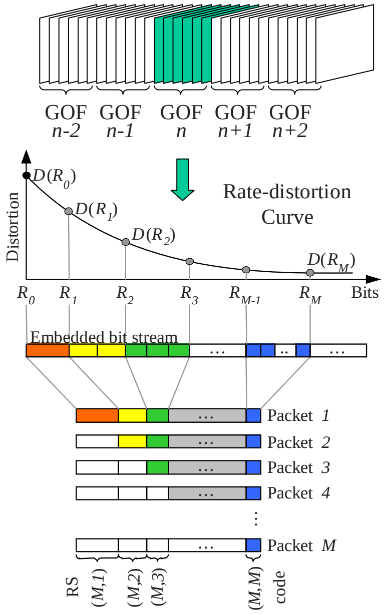
Multiple Description Coding

- Key feature: independent descriptions
  - No ordering of the descriptions
  - Any subset is decodable
  - In contrast with layered coding where loss of the base layer makes received enhancement layer useless
- State-of-the-Art MDC construction

  (Puri & Ramchandran '99, Mohr '00)
  - Uses layered coding and FEC as building blocks
  - Output M descriptions/packets

State-of-the-Art MDC

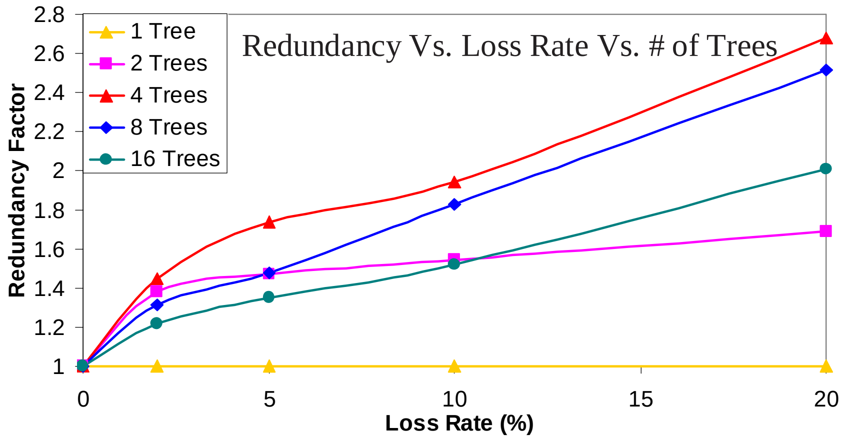
- Divide the stream into Group Of Frames (GOF)
- Prioritize the streaming data using *layered coding*
- For each GOF, apply different levels of *FEC* protection to data units depending on their importance
- FEC profile optimized given the packet size, # of descriptions M, and the distribution of the number of received descriptions/packets, p(m)
- Pure FEC, a *special* case of MDC where all clients have the same loss rate



Outline

- Problem statement
- State-of-the-art
- CoopNet approach to resilience:
  - Path diversity: multiple distribution trees
  - Data redundancy: multiple description coding
- Performance evaluation
- Summary and ongoing work

MDC Overhead



Processing: 1 Ghz CPU, 768MB RAM, Windows XP, C# implementation, < 20 ms for 1 second GOF

Tree Algorithms + MDC Evaluations: Methodology

- Use a flash crowd trace to simulate a streaming session
- Use real video clips as streaming data
- Perceived PSNR using averaged distortion across all clients, as the streaming quality metric

Flash Crowd Traces

- MSNBC streaming logs from Sep 11, 2001
  - Join time and session duration
  - Assumption: session termination ⇒ node stops participating
- Live streaming: 100 Kbps Windows Media Stream
  - up to ~18,000 simultaneous clients
  - ~180 joins/leaves per second on average
  - peak rate of ~1000 per second
  - ~70% of clients tuned in for less than a minute
    - high churn possibly because of flash crowd congestion

Video Data

Akiyo

Foreman

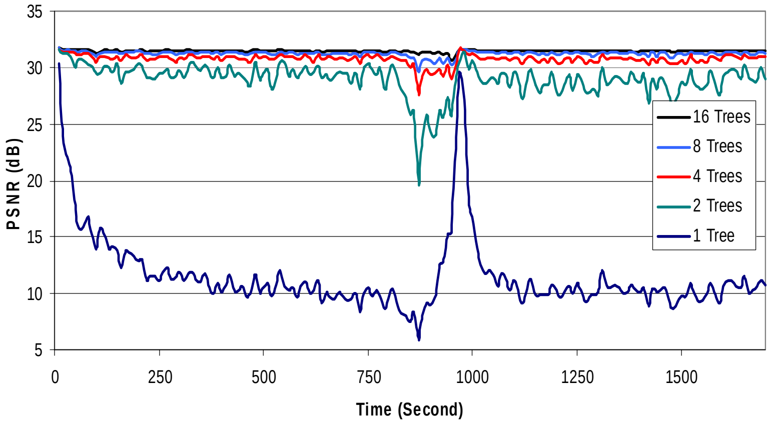
Stefan

- We don’t have the actual MSNBC video content
- Standard MPEG test sequences (10 seconds each)
- QCIF (176x144), 10 frames per second

Simulation Parameters

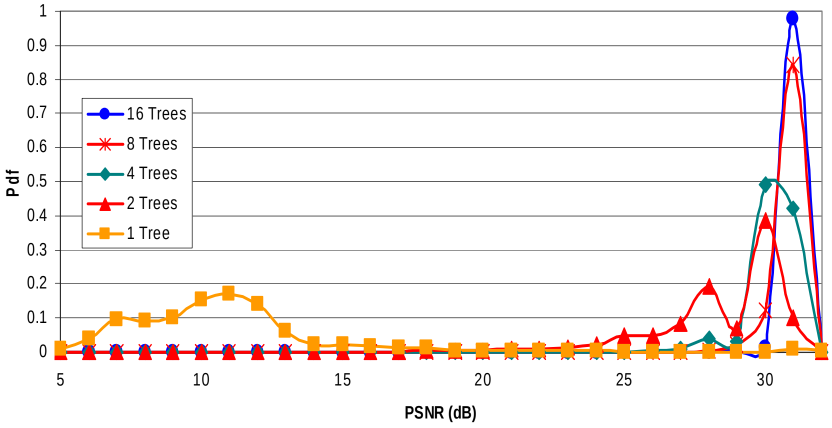
Server bandwidth: 20 Mbps
Peer bandwidth: 160 Kbps
Stream rate: 160 Kbps
Packet size: 1250 bytes
GOF duration: 1 second
# desciptions: 16
# trees: 1, 2, 4, 8, 16

Impact of Number of Trees

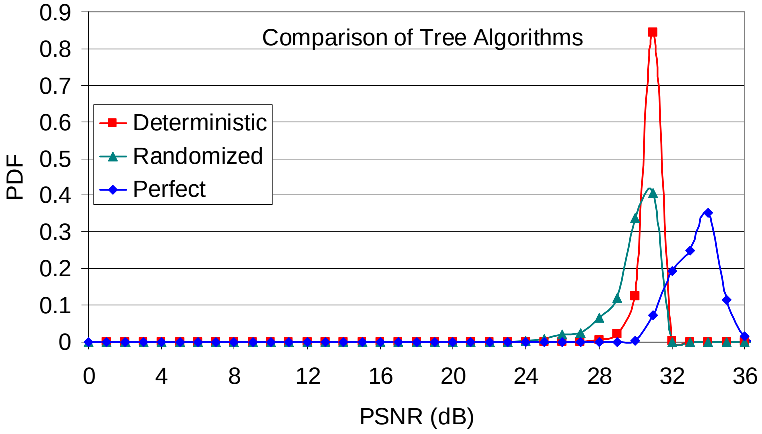


Multiple, diverse trees help significantly.
Much of the benefit is achieved with 8 trees.

Impact of Number of Trees

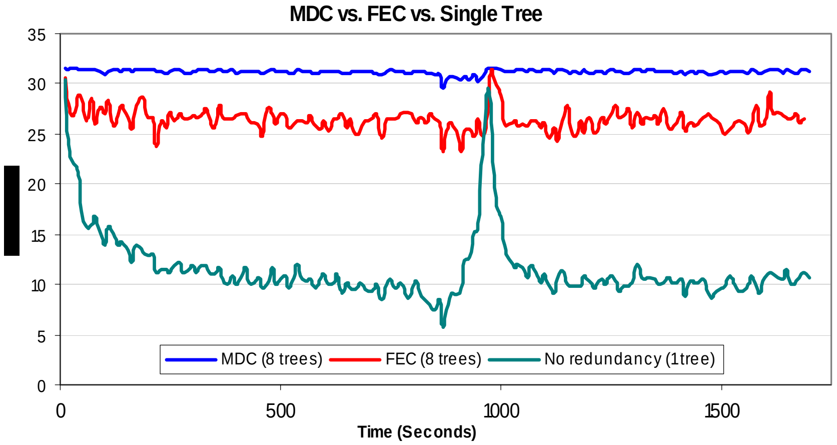


Randomized vs. Deterministic Tree Construction vs. Perfect Tree



Deterministic algorithm results in shorter trees that are less prone to disruption

MDC vs. Pure FEC



[redacted]

MDC is better able to adapt to a wide spatial distribution in packet loss than pure FEC.

Related Work

- Application-level multicast
  - ALMI [Pendarakis '01], Narada [Chu '00], Scattercast [Chawathe'00]
    - small-scale, highly optimized
  - Bayeux [Zhuang '01], Scribe [Castro '02]
    - P2P DHT-based
    - nodes may have to forward traffic they are not interested in
    - performance under high rate of node churn?
  - SplitStream [Castro '03]
    - layered on top of Scribe
    - interior node in exactly one tree ⇒ bounded bandwidth usage
- Infrastructure-based CDNs
  - Akamai, Real Broadcast Network, Yahoo Platinum
  - well-engineered network but for a price
- P2P CDNs
  - Allcast, vTrails

Related Work (Contd.)

- Coding and multi-path content delivery
  - Digital Fountain [Byers '98]
    - focus on file transfers
    - repeated transmissions not suitable for live streaming
  - Parallel downloads [Byers '02]
    - take advantage of lateral bandwidth
    - focus on speed rather than resilience
  - MDC for on-demand streaming in CDNs [Apostolopoulos '02]
    - what if last-mile to the client is the bottleneck?
  - Integrated source coding & congestion control [Lee '00]

Summary

- P2P streaming is attractive because it has the potential of being self-scaling
- Resilience to peer failures, departures, disconnections is a key concern
- CoopNet approach:
  - minimal demands placed on the peers
  - redundancy for resilience
    - multiple, diverse distribution trees
    - multiple description coding

Ongoing and Future Work

- Heterogeneity support:
  - Layered MDC
  - Congestion control framework
- More info: http://research.microsoft.com/projects/coopnet/
- Includes papers on:
  - case for P2P streaming: NOSSDAV '02
  - layered MDC: Packet Video '03
  - resilient P2P streaming: MSR Tech. Report
  - P2P Web content distribution: IPTPS '02

A typical client's experience on 911

Single-tree Distribution

CoopNet Distribution with FEC (8 trees)

CoopNet Distribution with MDC (8 trees)

Questions