Designing Overlay Multicast Networks for Streaming

Jevan Saks
Bruce Maggs
Konstantin Andreev
Adam Meyerson
The Problem

- Delivering streaming media to Media Servers
- Encoder → Media Server → User
- Centralized delivery model
- Need scalable and reliable solution
The Problem

- **Server bottlenecks**
  - Points of failure
  - Can only serve about 50Mbps of streams

- **Network bottlenecks**
  - Unpredictable topology
  - Best-effort delivery = No guarantees
Delivering Streaming Media

- **Server bottlenecks**
  - Points of failure
  - Can only serve about 50Mbps of streams

- **Network bottlenecks**
  - Unpredictable topology
  - Best-effort delivery = No guarantees
Live Streaming

Encoder -> Entry Point -> Reflectors -> Edge Servers (Sinks)
Proposed Solution

- A less centralized approach:
  
- **Entry Point**
  - Source of the stream, sends to reflectors

- **Reflector**
  - Intermediate server, can split and retransmit

- **Edge Server**
  - Reconstructs a better quality stream from what it receives and serves the end user
Approach

- Three-tier structure

Diagram:

- S – Sources
- R – Reflectors
- D – Sinks
Considerations

- Cost
- Capacity
- Bandwidth
- Quality
- Reliability
IP Parameters

- Success requirements ($\Phi$)
- Failure probabilities (p)
- Cost on edges (c)
- Cost on reflectors (r)
- Fanout restrictions (F)
Integer Program

- Indicator variables:
  - $z_i$ – reflector $i$ used
  - $y^k_i$ – stream $k$ sent to reflector $i$
  - $x^k_{ij}$ – stream $k$ sent to sink $j$ through reflector $i$
Constraints

- **Minimize:**
  \[
  \sum_{i \in R} r_iz_i + \sum_{i \in R} \sum_{k \in S} c_{ki}^k y_i^k + \sum_{i \in R} \sum_{k \in S} \sum_{j \in D} c_{ij}^k x_{ij}^k
  \]

- **Subject To:**
  1. \( y_i^k \leq z_i \)
  2. \( x_{ij}^k \leq y_i^k \)
  3. \( \sum_{k \in S} \sum_{j \in D} x_{ij}^k \leq F_i z_i \)
  4. \( \sum_{j \in D} x_{ij}^k \leq F_i y_i^k \)
  5. \( \sum_{i \in R} x_{ij}^k w_{ij}^k \geq W_j^k \)
  6. \( x_{ij}^k \in \{0, 1\}, y_i^k \in \{0, 1\}, z_i \in \{0, 1\} \)
Solving the IP

- ILOG Cplex (concert library)
- DashOptimization Xpress-MP
- GLPK (GNU Linear Programming Kit)
An Example (Thanks to graphviz)
Approximation

- **Why?** IP is slow, and topology is large and changes quickly

- **Randomized rounding:**
  - Cost violated by factor $O(\log n)$
  - Fanout/weight constraints violated by factor of at most 2

- **Modified GAP Assignment**
  - Uses max-flow on a graph of size $O(|R| \cdot |D|)$
  - Cost violated again by $O(\log n)$
  - Fanout/weight constraints violated by another factor of at most 2
Comparison

IP Solution (Cost = 51)  Approx Solution (Cost = 52)
In Progress

- Multi-source approximations
- Timing comparisons
- Violations in “average-case” scenarios
Extensions

- Capacities on all edges
- Capacities from reflector-edgeserver
- Bandwidth requirements
- Color constraints
Extension’s Constraints

- Color constraints

\[ \sum_{i \in R_{\ell}} x_{i,j}^k \leq 1 \quad \forall j \in D, k \in S, \ell \in [m] \]

where \( R = R_1 \cup R_2 \cup \ldots \cup R_m \)

\( (R_i \) is the set of reflectors on the \( i^{th} \) ISP)\n
7. Here different colors represent different ISPs

8. There is no added value in serving to a fixed sink from two reflectors of the same color.
Future Work

- Use real-world data from Akamai
- Implement extensions
- Improve approximation?
Best Case Scenario?