Edge-based Transcoding for Adaptive Live Video Streaming

Pradeep Dogga*  Sandip Chakraborty†  Subrata Mitra‡  Ravi Netravali*

UCLA *  Adobe †  ‡
Increase in live video streaming
Live Video Streaming Architecture

Goal: Video as live as possible

Network Contention!

Transcoding Servers

CDN

Compute Contention!

Quality

- 720p
- 480p
- 360p

Viewers

- 720p
- 360p

Concert

Edge Router

1080p

Liveness lost!
Problem

• Sharing infrastructure → Video liveness lost

• Key Question: Can we alleviate the problem by incorporating sources?
  • Can we use a P2P approach for transcoding and transmitting live video?
Reason 1

- Existing live video applications do not adapt to network conditions!
Reason 2

- Phones are energy constrained

Codecs are energy efficient!
Reason 3

- Transcoding with smartphones should keep up with live video time limits

<table>
<thead>
<tr>
<th>Device</th>
<th>1080p -&gt; 720p</th>
<th>720p -&gt; 360p</th>
<th>1080p -&gt; 360p</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTC U11</td>
<td>15ms</td>
<td>7ms</td>
<td>9ms</td>
</tr>
<tr>
<td>Pixel 2 XL</td>
<td>8ms</td>
<td>6ms</td>
<td>9ms</td>
</tr>
<tr>
<td>PC (FFMPEG)</td>
<td>6ms</td>
<td>2ms</td>
<td>2ms</td>
</tr>
</tbody>
</table>

Codecs are fast!

Time taken to transcode video per frame
Peer to peer transcoding and streaming

Video Source

1080p → 720p

720p → 480p

1080p

720p

480p
Balancing Trees

- Balancing trees when nodes leave or enter the system to minimize latency
- Restructuring trees for fairness in energy and available bandwidth consumption
- Can be modelled as an optimization problem
Details - Registration

1080p → 720p

720p → 480p

Video Source

“join 1080p”
“Contact for 1080p”

1080p

720p

480p
Details – Failure/Departure of viewer

Video Source

1080p → 720p
Repeat registration

720p
Repeat registration

720p → 480p

1080p

480p
Details – Swapping transcoders

- Video Source
  - 1080p → 720p
  - 720p → 480p

- 1080p
- 720p
- 480p
Open Issues

• **Peer Presence**: Incentives for transcoding and transmitting
• **Data Integrity**: Overheads in verification and attestation
• **Seeks and Delayed Viewing**: Mechanism to store stream intermittently
• **Viewer Flux**: Need decentralized heuristic based optimization to closely approximate to optimal distribution of viewers
Alternate use-cases

• Real-time Video Analytics can avoid server-side computation overheads
• Private Live Video can be enabled without additional infrastructure
Conclusion

• Significant opportunity to better use client devices in live streaming systems
  • Hardware accelerated codecs on phones are fast and energy efficient

Thank you!

dogga@cs.ucla.edu