AIO-TFRC: A light-weighted rate control scheme for streaming over wireless

Presented by Wei Wei

Minghua Chen and Avidah Zakhor

Video and Image Processing Lab, Department of EECS
University of California at Berkeley
Outline

- Motivation
- AIO-TFRC
- Simulations
- Conclusions and discussions
Motivation: Rate Control for Wireless Video

- Streaming Rate control is important for both wired and wireless networks:
  - @ what rate should the video be streamed?
  - Goals:
    - Fully utilize bottleneck bandwidth with minimum packet loss rate
    - Fairness between applications

- Widely accepted solution TFRC (TCP-friendly Rate Control) assumes packet loss is only caused by congestion ➔ wire-line Internet

- Wireless physical channel also causes packet loss:
  - Hard to distinguish between congestion and physical layer loss
  - TFRC fails to work
  - In 1xRTT CDMA network, TFRC achieves 56% utilization on wireless

<table>
<thead>
<tr>
<th></th>
<th>Wired</th>
<th>Wireless</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>TCP</td>
<td>-</td>
</tr>
<tr>
<td>Video</td>
<td>TFRC</td>
<td>?</td>
</tr>
</tbody>
</table>
Two classes of approaches

- Differentiating between packet loss due to congestion and that due to physical channel error
  - Only take congestion based loss into account
  - New problem → old problem; apply traditional wired-line rate control

- Changes needed to the infrastructure
  - Balakrishnan et.al. 96, Ratnam and Matta 03, Chio et. al. 02, Ding and Jamalipour 01, Cobb and Agrawal 95, Chiasserini and Meo 01, Rendon et. al. 02, …

- Explicit Loss Notification (ELN) based
  - Balakrishnan and Katz 98, Yang et. al. 01, …

- End-to-end statistics based
  - Samaraweera 99, Biaz and Vaidya 99, Lee et. al. 03, Sinha et. al. 99, Cen et. al. 03, Yang et. al. 04, Yang and Qian et. al. 04, Akan and Akyildiz 04, Tang et. al. 01, Liu et. al. 03, Parsa et. al. 99, …

- Our approach [Chen & Zakhor InfoCom 2004]: Do not differentiate, open multiple connections
Previous approach - MULTFRC

Inversely Increase and Additively Decrease (IIAD) on number of connections:

\[
n = \begin{cases} 
  n - \beta, & \text{if } \text{ave}_\text{rtt} - \text{rtt}_\text{min} > \gamma \text{rtt}_\text{min}; \\
  n + \alpha/n, & \text{otherwise.}
\end{cases}
\]

Empirically choose \(\alpha=\beta=1\), and \(\gamma=0.2\)

Experimental Results over 1XRTT network:

<table>
<thead>
<tr>
<th>scheme</th>
<th>throughput (kbps)</th>
<th>rtt (ms)</th>
<th>packet loss rate</th>
<th>ave. # of conn.</th>
</tr>
</thead>
<tbody>
<tr>
<td>one TFRC</td>
<td>54</td>
<td>1624</td>
<td>0.031</td>
<td>N/A</td>
</tr>
<tr>
<td>MUI TFRC</td>
<td>86</td>
<td>2512</td>
<td>0.045</td>
<td>1.8</td>
</tr>
</tbody>
</table>
Two Drawbacks of MULTFRC

- **Quantization effects**
  - # of connections must be integer ➔ quantization effects
  - Results in unnecessarily low utilization when # of connections is small

- **Overhead of operating multiple TFRC connections**
  - Overhead in system resource and signaling
  - Not good for resource-limited mobile handheld

- **Our approach**: combine multiple TFRC connections into one: AIO-TFRC!
Outline

- Motivation
- AIO-TFRC
- Simulations
- Conclusions and discussions
All-In-One TFRC (AIO-TFRC)

○ Basic idea:
  • Physically open one connection
    – Reduce overhead
  • Avoid the quantization on the number of connections
    – Remove quantization effect

○ Specifically:
  • Control the virtual number of connections, $n$, in the same way as MULTFRC
    
    $$n = \begin{cases} 
    n - \beta, & \text{if } \text{ave}_\text{rtt} - \text{rtt}_\text{min} > \gamma \text{rtt}_\text{min}; \\
    n + \alpha/n, & \text{otherwise.}
    \end{cases}$$
  
  • Empirically choose $\alpha = \beta = 1$, and $\gamma = 0.5$
  • Make one connection send at a rate $n$ times of that of one TFRC by TFRC+BFLD (Bandwidth Filtered Loss Detection)

Naïve approach

- Sender side sends @ rate
- Receiver side measures the packet loss event rate, \( p \), based on all the arriving packets

\[
T = n \times \frac{k \cdot S}{\text{rtt} \sqrt{p}}
\]

Loss event rate of one TFRC connection

Loss Event Interval = 8–2 = 6

Loss Event Rate = 1/6

RTT

Loss Event Interval = 15–3 = 12

Loss Event Rate = 1/12

Underestimates \( p \) \( \Rightarrow \) Inflates sending rate > \( n \) times of that of one TFRC
Make one connection send at a rate \( n \) times of that of one TFRC: **TFRC+BFLD**

- **Sender side**
  - Set sending rate
  \[
  T = n \times \frac{k \cdot S}{rtt \cdot \sqrt{p}}
  \]
  - Evenly mark the headers of selected data to form a virtual single TFRC flow
    - \( 1/n \) of all outgoing packets are marked

- **Receiver side** measures the packet loss event rate, \( p \), based on the **virtual TFRC flow** with marked packets.

![Diagram showing packet events and loss intervals](image)
Outline

- Motivation
- AIO-TFRC
- Simulations
- Conclusions and discussions
**NS-2 Simulation Setting**

- $B_w = 1$ Mbps
- $p_w$ varies from 0.00 - 0.08 in increments of 0.02
- Wireless link is simulated using a wired link + exponential error model
- Packet size $S=1000$ bytes
- Measure throughput every 10 seconds, end-to-end packet loss rate every 30 seconds, average rtt every 100 packets
Quantization effects are eliminated

- Better bandwidth utilization

![Graphs showing quantitative data](image-url)
Fairness among AIO-TFRC flows

- Fairness Ratio = individual flow’s throughput / average throughput
- 1 is perfect

<table>
<thead>
<tr>
<th>receiver</th>
<th>fairness ratio $p_w=0.01$</th>
<th>fairness ratio $p_w=0.04$</th>
<th>receiver</th>
<th>fairness ratio $p_w=0.01$</th>
<th>fairness ratio $p_w=0.04$</th>
</tr>
</thead>
<tbody>
<tr>
<td>d1</td>
<td>1.00</td>
<td>0.99</td>
<td>d9</td>
<td>1.02</td>
<td>1.03</td>
</tr>
<tr>
<td>d2</td>
<td>0.99</td>
<td>0.99</td>
<td>d10</td>
<td>0.98</td>
<td>0.98</td>
</tr>
<tr>
<td>d3</td>
<td>0.96</td>
<td>1.00</td>
<td>d11</td>
<td>0.97</td>
<td>1.01</td>
</tr>
<tr>
<td>d4</td>
<td>0.99</td>
<td>0.97</td>
<td>d12</td>
<td>0.99</td>
<td>0.99</td>
</tr>
<tr>
<td>d5</td>
<td>1.05</td>
<td>0.95</td>
<td>d13</td>
<td>0.99</td>
<td>1.01</td>
</tr>
<tr>
<td>d6</td>
<td>1.04</td>
<td>1.01</td>
<td>d14</td>
<td>1.01</td>
<td>0.97</td>
</tr>
<tr>
<td>d7</td>
<td>1.03</td>
<td>1.02</td>
<td>d15</td>
<td>1.01</td>
<td>0.98</td>
</tr>
<tr>
<td>d8</td>
<td>1.02</td>
<td>1.03</td>
<td>d16</td>
<td>0.96</td>
<td>1.02</td>
</tr>
</tbody>
</table>

$B_w = 5$Mbps, $RTT_{min} = 503ms$
Fairness between AIO-TFRC and TCP: A Moot point

TCP underutilizes, AIO-TFRC fully utilizes $\Rightarrow$ fairness is ill defined $\Rightarrow$ can not compare the throughputs directly

**Optimistic fairness**: existence of AIO-TFRC does not affect TCP’s throughput significantly

<table>
<thead>
<tr>
<th>settings</th>
<th>8 AIO-TFRC + 8 TCP</th>
<th>16 TCP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ave. thput. (AIO-TFRC) (kbps)</td>
<td>ave. thput. (TCP) (kbps)</td>
</tr>
<tr>
<td>$p_w=0.01$ $\gamma=0.5$</td>
<td>436.501</td>
<td>168.048</td>
</tr>
<tr>
<td>$p_w=0.01$ $\gamma=0.1$</td>
<td>379.656</td>
<td>185.286</td>
</tr>
<tr>
<td>$p_w=0.02$ $\gamma=0.5$</td>
<td>486.821</td>
<td>120.313</td>
</tr>
<tr>
<td>$p_w=0.02$ $\gamma=0.1$</td>
<td>449.226</td>
<td>130.953</td>
</tr>
</tbody>
</table>
Outline

- Motivation
- AIO-TFRC
- Simulations
- Conclusions and discussions
Globally Converge to Stable Point?

Question: Is it possible to achieve highest throughput and lowest packet loss rate, using an end-to-end solution?

Answer: MULTFRC or AIO-TFRC

AIO-TFRC outperforms MULTFRC in that:
- No quantization effect → higher bandwidth utilization
- Opening one connection → less operational overhead → good for actual implementation in resource-limited handheld device

Limitation: need to detect route change to update the propagation delay estimate, i.e. \( \text{rtt}_{\text{min}} \)

Future work: what if widely deployed?
- Both the virtual number of connections and sending rate of each connection are dynamically varied
- Will network converge to stable and optimal point?
- Will the solution scale to a large network?
Simulations and Experiments

○ NS-2 Simulations

○ Actual experiments over Verizon 1xRTT CDMA datanetwork

<table>
<thead>
<tr>
<th>scheme</th>
<th>throughput (kbps)</th>
<th>rtt (ms)</th>
<th>packet loss rate</th>
<th>ave. # of conn.</th>
</tr>
</thead>
<tbody>
<tr>
<td>one TFRC</td>
<td>54</td>
<td>1624</td>
<td>0.031</td>
<td>N/A</td>
</tr>
<tr>
<td>MULTFRC</td>
<td>86</td>
<td>2512</td>
<td>0.045</td>
<td>1.8</td>
</tr>
</tbody>
</table>