Dynamic Adaptive Streaming Solutions for Future Networks: HTTP/2 as a Key

Thesis of:
Mariem BEN YAHIA
started on: January 4, 2016

Supervised by:
Gwendal SIMON - Loutfi NUAYMI (IMT Atlantique Rennes)
Yannick LE LOUEDECU (Orange Labs Lannion)

CSID comitee members
Bernard COUSIN (Telecom-Paristech- IRISA)
Jean-Claude DUFOURD (Telecom-Paristech)
Plan

1. Introduction: the objectives of my thesis
2. HTTP/2 protocol
3. My first contribution
4. My second contribution
5. Conclusions and discussion
Introduction: the objectives of my thesis

Title

Dynamic Adaptive Streaming Solutions for Future Networks: HTTP/2 as a Key

Task 1
Define future streaming scenarios

Task 2
Analyze the issues of the current streaming solutions

Task 3
Propose HTTP/2-based solutions
HTTP/2 protocol

HTTP/2: transforms the communication into an exchange of binary encoded frames. HTTP/2 frames convey messages that belong to a particular stream and all streams are multiplexed within a single TCP connection.
HTTP/2 protocol
HTTP/2 multiplexing illustration
## HTTP/2 VS HTTP/1

<table>
<thead>
<tr>
<th>Options</th>
<th>HTTP/1</th>
<th>HTTP/2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stop an ongoing request/response communication</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Set dependency between request/response communications</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Change dependency of request/response communications</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Set weights between request/response communications</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Change weights of request/response communications</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Objective: Propose dynamic adaptive streaming solutions for conventional videos with about one second delivery latency*. * Latency between video frame streaming and video frame display

Plan:
1. Problems of DASH current solutions
2. Video encoding
3. Our proposed HTTP/2-based solution
4. Datasets
5. Results
6. Conclusion
Dynamic adaptive streaming

Reference: Orange France
Problems of current solutions

**DASH**: The client estimates the bandwidth and selects for each video segment the quality that best matches to the estimated bandwidth.

**Bandwidth estimation**: For small buffer scenarios, DASH uses throughput-based estimation mechanism.

**DASH errors**: We statistically analyze the deficit between video segment bit-rates and real throughputs. ¹

<table>
<thead>
<tr>
<th>Segment duration</th>
<th>% of seconds with deficit</th>
<th>% of deficit amplitude</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 s</td>
<td>23.5</td>
<td>23.8, 17.7</td>
</tr>
<tr>
<td>4 s</td>
<td>23.1</td>
<td>24.8, 18.9</td>
</tr>
<tr>
<td>6 s</td>
<td>19.0</td>
<td>25.0, 22.7</td>
</tr>
<tr>
<td>8 s</td>
<td>19.0</td>
<td>25.0, 21.2</td>
</tr>
<tr>
<td>10 s</td>
<td>15.4</td>
<td>26.3, 18.2</td>
</tr>
</tbody>
</table>

Table 1. Throughput deficit on DASH experimentation

¹We use available 4G network traces to analyze DASH behaviors on low latency delivery.
Dependencies of encoded video frames

Different types of video frames

Each video frame has:
- Display time
- Decoding deadline
- Importance to the whole video
- Dependencies
Video frame discarding: consists in dropping the least important frames of the encoded video content in constrained networks.

Complex implementation:

■ Mehdian and Liang [2014] propose a recursive optimal video frame discarding model.

■ Darabkh et al. [2015] propose to drop video frames at the network layer. They modify the UDP header field to represent the video frame importance.

■ Gangadharan et al. [2011] drop video frames at the receiver buffer side.
Delivery order of encoded video frames

**Figure:** Encoding video frame dependencies

**Figure:** Best video frame delivery order
Our proposed solution: scheme

Objective: Cancel the ongoing delivery of the less important video frames of a DASH video segment.

How?: Use HTTP/2 to dynamically control the video frame delivery at the applicative network layer: **Send each video frame in an HTTP/2 stream.**

1) Delivery order of video frames

2) Set dependency between HTTP/2 streams
We opt for the FCFS implementations of the HTTP/2 priority feature.

Size (frames): 2  2  4  1  2  2  2  1  1

Weights have no importance
Only dependency is important

HTTP/2 First Come First Served (FCFS) prioritization strategy
Main principles of our proposal

An entirely client-based scheme:

- Analyze video structure and extract meta-data
- Associate each video frame to an HTTP/2 stream
- Compute the importance of every video frame
- Decide which HTTP/2 stream to cancel
Our proposed algorithms: optimal ILP

**Optimal algorithm** Integer linear programming: Decides the optimal delivery of video frames considering their importance

Goal: Find an upper bound of the achievable performance

\[
\begin{align*}
\text{max} & \quad \sum_v g^v y^v \\
\text{s.t} & \quad \sum_{h \in H} x^v_h = s^v y^v \quad (1) \\
& \quad \sum_{v \in V} x^v_h \leq 1 \quad (2) \\
& \quad y^{vc} \leq y^v \quad (3) \\
& \quad \text{if } t^v_{\text{display}} \leq t^v_{\text{vp}} \text{ then } \quad t^h_{\text{rx}} x^v_{hp} \leq t^v_{\text{vp}} (1 - y^v) + y^v t^v_{\text{display}} \quad (4) \\
& \quad \text{if } t^h_{\text{rx}} \geq t^v_{\text{display}} \text{ then } \quad x^v_h = 0 \quad (5) \\
\end{align*}
\]

With
\[
\begin{align*}
& y^v \in \{0, 1\} , x^v_h \in \{0, 1\} , \\
& v \in V, h \in H, v_c \in C^v , v_p \in P^v
\end{align*}
\]
Our proposed algorithms: heuristics

R-R algorithm Detects all late video frames at each HTTP/2 frame reception and cancels their delivery

R-R-T algorithm Detects all late video frames periodically (each T time) and cancels their delivery

FIFO algorithm (reference) All video frames are delivered in a FIFO order (no canceling)
Network datasets: Capture TCP packets on different network types: cellular and WiFi networks.
Simulation datasets: videos

**Video datasets**: Include dynamic and static videos. We consider a 6-second video segment for each video.
Compare the traffic overhead resulting from requesting each video frame in an HTTP/2 stream to the traffic overhead resulting of requesting the overall video segment in a single stream: **DL overhead increasing from 4.34% to 5.76%**.

<table>
<thead>
<tr>
<th></th>
<th>Download</th>
<th>Upload</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>multi-stream</td>
<td>mono-stream</td>
<td>multi-stream</td>
</tr>
<tr>
<td></td>
<td>Size</td>
<td>Number</td>
<td>Size</td>
</tr>
<tr>
<td>Uncompressed header</td>
<td>114</td>
<td>1</td>
<td>114</td>
</tr>
<tr>
<td>Compressed headers</td>
<td>15</td>
<td>179</td>
<td>0</td>
</tr>
<tr>
<td>Data Frame headers</td>
<td>8</td>
<td>180</td>
<td>8</td>
</tr>
<tr>
<td>Settings</td>
<td>2</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>Windows_update</td>
<td>4</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td><strong>Total overhead</strong></td>
<td><strong>5.76%</strong></td>
<td></td>
<td><strong>4.34%</strong></td>
</tr>
</tbody>
</table>

Table. HTTP/2 protocol overheads for video streaming
Results: Spatial distortion

PSNR enhanced on dynamic videos over both networks. No real difference is observed on the PSNR for static videos.

(a) dynamic videos with cellular networks

(b) static videos with cellular networks

(c) dynamic videos with WiFi networks

(d) static videos with WiFi networks

Fig. Cloud of points representing the average PSNR of dynamic or static videos combined to cellular or WiFi networks
Results: Spatial distortion - Impact of time $T$

![Graph showing PSNR comparison with R-R-T algorithms for dynamic videos combined with all network traces.](image)

Fig. PSNR comparison of R-R-T algorithms for dynamic videos combined with all network traces.
A jitter is a display discontinuity due to missing video frames.
- Small jitter: 3 to 5 consecutive video frames are missing
- Medium jitter: 6 to 15 consecutive video frames are missing
- Long jitter: 15 to 40 consecutive video frames are missing
- Crash: more than 40 consecutive video frames are missing
Temporal distortion: frequency and duration

Fig. Temporal distortion per algorithm for video/network combinations which we classify into two categories, a first category for whom the basic FIFO does not crash and a second category for whom the basic FIFO crashes.
The proposed algorithms enable video delivery with one-second delivery latency
The proposed algorithms reduce the spatial and temporal distortion due to missing video frames
Our HTTP/2-based solution slightly increases the DL overhead but it gives the client more flexibility than mono-HTTP/2 stream solutions.

Publications:
Workshop paper: Infocom conference, US, 2017
Review paper: ACM TOMM journal 2018 (submitted + first revision)
Intensive subjective tests should be driven to evaluate the QoE.

- The results of our objective metrics motivate to drive subjective tests.

Study the benefits of intelligent decoder implementations.

- We modeled a second ILP (Decoder 2) of an intelligent decoder implementation that accept frames that arrive too early for the decoding of successive frames.

- We present primary results in Tab.1 but a complete study should be done.

<table>
<thead>
<tr>
<th>Initial buffer</th>
<th>0,1</th>
<th>0,2</th>
<th>0,5</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Decoder 1</td>
<td>Decoder 2</td>
<td>Decoder 1</td>
</tr>
<tr>
<td>Displayed Importance</td>
<td>35,7%</td>
<td>76,9%</td>
<td>35,7%</td>
</tr>
<tr>
<td>Received I</td>
<td>50%</td>
<td>50%</td>
<td>50%</td>
</tr>
<tr>
<td>Received P</td>
<td>25%</td>
<td>75%</td>
<td>25%</td>
</tr>
<tr>
<td>Received B</td>
<td>38%</td>
<td>59%</td>
<td>38%</td>
</tr>
</tbody>
</table>

Tab.1 Comparison between basic (decoder 1) and intelligent (decoder 2) decoding implementations
My second contribution

Objective: Propose adaptive streaming solutions for immersive videos based on HTTP/2.

Plan:
1. Problems of current solutions
2. Tiled video
3. Our proposed HTTP/2-based solution
4. Primary results
5. Ongoing works
Problems of current solutions

Viewport: At any given time, the client only watches a part of the video called **viewport**.

Bandwidth consumption: Streaming the entire video at high quality is a waste of resources.

QoE: Streaming the entire video at a single quality reduces the selected quality of the viewport.
Tiled viewport dependent streaming: Spatially divide the video into independent parts called tiles viewport.

Viewport-dependent solutions: Deliver high quality only for tiles overlapping with the viewport.

- **Benefits:** Reduce the BW consumption and enhance the QoE but depend on viewport prediction.
- **Challenges:** Viewport-prediction is prone to errors and its accuracy increases over time.

**Our goal:** Model streaming solutions considering the bandwidth consumption and the viewport prediction errors in constrained networks.
Tiled immersive video & viewport prediction

**Figure:** Real viewport projection on tiled media (visible tiles)

**Figure:** Predicted viewport projection on tiled media (predicted visible tiles)

**Our proposed scheme:** Request each tile in a HTTP/2 stream and manage the tiles delivery over time according to the evolution of the prediction accuracy.
Viewport accuracy schemes

- **Big errors**
  - Prediction 1

- **No errors**
  - Prediction 2

- **Small errors**
  - Display time

- **Medium errors**
  - Video frames of segment $k + 1$

**Big, medium or small errors for all video frames**

- **Small or nonexistent errors for all video frames**
  - Video frames of segment $k + 1$

- **Display time**

---

Dynamic Adaptive Streaming Solutions for Future Networks: HTTP/2 as a Key - Mariem BEN YAHIA
Our proposed solution

**Quality selection**: Maximize the quality levels of extended visible tiles and send the rest of tiles with low quality level

**Two viewport predictions**: The client requests the quality levels of tiles at the first prediction and then review its decisions at the second prediction

**Viewport-dependent priority**: The client requests each tile in an HTTP/2 stream. It attributes the stream weights in order to prioritize the delivery of visible tiles.

**HTTP/2 decisions**: At the second prediction the client can cancel a tile delivery and/or requests it in a new quality level

**In constrained networks**: At the second prediction the client can cancel the ongoing delivery of tiles that are on the opposite side of the viewport
HTTP/2 features

**HTTP/2**: Tiles are independent. We opt for the RR priority implementation.

*Figure*: HTTP/2 streams share concurrently the same resources according to their weight.
HTTP/2 DASH client decisions

- Extended viewport: viewport + an extension of $10^\circ \times 10^\circ$ as a safety margin.
- Background: the set of pixels out of the extended viewport.
- Opposite viewport: the symmetric image of the viewport depending on the center of the sphere.

<table>
<thead>
<tr>
<th>Tiles position</th>
<th>Tile completely received</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prediction 1</td>
<td>Prediction 2</td>
</tr>
<tr>
<td>Extended Viewport</td>
<td>Extended Viewport</td>
</tr>
<tr>
<td>Background</td>
<td>Extended Viewport</td>
</tr>
<tr>
<td>Extended Viewport</td>
<td>Background</td>
</tr>
<tr>
<td>Any</td>
<td>Opposite viewport</td>
</tr>
</tbody>
</table>

**Table:** Actions of the HTTP/2 client at the second prediction
Primary results: Status of the visible pixels

i) Small errors

ii) Medium errors

iii) Big errors

a) 1 prediction HTTP1

b) 2 predictions HTTP1

c) 2 predictions HTTP2 viewport weight
Primary results: Bandwidth consumption

i) Small errors

ii) Medium errors

iii) Big errors

- 1 prediction HTTP1
- 2 predictions HTTP1
- 2 predictions HTTP2 viewport weight
Primary results: Impact of the weight function.

i) Small errors
ii) Medium errors
iii) Big errors

Bandwidth

Viewport ratio

Unreceived
Low quality

c) 2 predictions HTTP2 default weight

c) 2 predictions HTTP2 sized weight

c) 2 predictions HTTP2 viewport weight
Conclusion

Our proposed viewport adaptive HTTP/2 streaming scheme:

- enhances the quality of visible pixels.

- reduces the ratio of unreceived visible pixels in constrained networks.

- automatically adapts to the available bandwidth (it cancels the opposite viewport only if the network is challenging)

- strongly depends on the HTTP/2 weight function
Ongoing works

- Study the HTTP/2 related traffic overhead
- Study the impact of RTT on the performance of our proposed algorithms
- Enrich the tests by varying the video motion type and the tile number
- Evaluate the QoE with PSNR or VMAF

Publications:
Conference: ISM proceedings, Taiwan, December 2018
(submission on July, 20)
Perspectives

- Study the effect of several quality levels selection on the performance of our proposed solution
- Optimize the time of the second prediction depending on the bandwidth state and the history of the viewport accuracy
General conclusion

- We show the benefits of the HTTP/2 priority feature for video delivery with low latency requirements

- The performance of the HTTP/2-based solution strongly depends on the priority implementation strategy

- Optimizing the video delivery through the network strongly depends on the encoding techniques of the content

Perspective

- Study and model HTTP/2-based streaming solutions for Scalable Video Coding (SVC)
Annexe 0

<table>
<thead>
<tr>
<th>Input</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$H$</td>
<td>Set of HTTP/2 frames successfully received by the client</td>
</tr>
<tr>
<td>$V$</td>
<td>Set of video frames</td>
</tr>
<tr>
<td>$P^v$</td>
<td>Set of video frames on which the video frame $v$ depends</td>
</tr>
<tr>
<td>$C^v$</td>
<td>Set of video frames depending on video frame $v$</td>
</tr>
<tr>
<td>$s^v$</td>
<td>Number of HTTP/2 frames needed to transmit video frame $v$</td>
</tr>
<tr>
<td>$t^v_{display}$</td>
<td>Display time of video frame $v$</td>
</tr>
<tr>
<td>$g^v$</td>
<td>Relative importance of video frame $v$ on the entire video quality</td>
</tr>
<tr>
<td>$h_{size}$</td>
<td>HTTP/2 frame size</td>
</tr>
<tr>
<td>$v_{size}$</td>
<td>Video frame $v$ size</td>
</tr>
<tr>
<td>$r^h_{rx}$</td>
<td>Arrival time of HTTP/2 frame $h$ at the client side</td>
</tr>
<tr>
<td>$M$</td>
<td>Display time of the latest video frame of the GOP</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Decision</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_h^v$</td>
<td>equal to 1 if HTTP/2 frame $h$ contains data relative to video frame $v$ and 0 otherwise</td>
</tr>
<tr>
<td>$y^v$</td>
<td>equal to 1 if video frame $v$ is displayed ($i.e.$, received as well as all its parents before $t^v_{display}$) and 0 otherwise</td>
</tr>
<tr>
<td>$z^v$</td>
<td>equal to 1 if video frame $v$ is received at any time at the client side, and 0 otherwise (only used in the case of hypothesis H2)</td>
</tr>
</tbody>
</table>

**Table:** Notations for inputs and decision variables
### Decoder 1 implementation

\[
\text{max} \quad \sum_v g^v y^v \\
\text{s.t} \quad \sum_{h \in H} x^h_v = s^v y^v \tag{6} \\
\sum_{v \in V} x^h_v \leq 1 \tag{7} \\
y^v_{vc} \leq y^v \tag{8} \\
\text{if } t^v_{\text{display}} \leq t^v_{\text{display}} \\
\text{then } t^h_{rx} x^v_{h} \leq t^v_{\text{display}} (1 - y^v) + y^v t^v_{\text{display}} \tag{9} \\
\text{if } t^h_{rx} \geq t^v_{\text{display}} \\
\text{then } x^v_{h} = 0 \tag{10} \\
\text{With } y^v \in \{0, 1\}, x^v_{h} \in \{0, 1\}, \\
v \in V, h \in H, v_c \in C^V, v_p \in P^V
\]

### Decoder 2 implementation

\[
\text{max} \quad \sum_v g^v y^v \\
\text{s.t} \quad \sum_{h \in H} x^h_v = s^v z^v \tag{11} \\
\sum_{v \in V} x^h_v \leq 1 \tag{12} \\
y^v \leq z^v, v_p \in P^v \cup \{v\} \tag{13} \\
x^h_v t^h_{rx} \leq t^v_{\text{display}} + M(1 - y_{vc}), v_c \in C^V \cup \{v\} \tag{14} \\
\text{With } z^v \in \{0, 1\}, y^v \in \{0, 1\}, x^v_{h} \in \{0, 1\}, \\
v \in V, h \in H, v_c \in C^V, v_p \in P^V
\]