SmarterPhones: Anticipatory Download Scheduling for Wireless Video Streaming

Martin Dräxler*, Johannes Blobel*, Philipp Dreimann*, Stefan Valentin+ and Holger Karl*

*University of Paderborn, Germany
+Bell Labs, Alcatel Lucent, Stuttgart, Germany

NetSys 2015
10.03.2015
Overview

- Project “Smarter Phones and Networks” (SPAN)
  - Cooperation between UPB and Alcatel Lucent Bell Labs, Stuttgart
  - 2 student project groups
  - Project “CROWD” (FP7)
“Traditional” video streaming

- Single source file transmitted continuously
- Transmitted over unreliable datagram protocol

- Packet loss, drops in data rate
  → Distortion, artifacts, playback stops

- Multiple quality levels
  → Complex approaches (e.g. SVC)

- QoE metrics
  → MOS, PSNR, ...

Segmented video streaming over HTTP

- Source file divided into segments of a typically fixed duration
- Segments sequentially downloaded via standard HTTP

- Packet loss, data rate drops $\rightarrow$ playback stops
  - Download segment i+1 before segment i has finished playback
  - Segments can be buffered (=downloaded earlier)

- Multiple quality levels $\rightarrow$ provide segments in different qualities
  - Different resolutions, encodings, ...

- QoE metrics $\rightarrow$ easily observable
  - Playback interruptions
  - Delivered quality level

- HTTP Live Streaming (HLS), MPEG DASH
- Android, iOS, …
Data Rate Prediction

• Channel prediction
  • Accurately predict channel conditions → easily derive data rate
  • Very (very) short time scales → milliseconds
  • Used for radio resource scheduling

• User movement & behavior prediction
  • Very different scopes → next cell/BS, trajectory, path
  • Different time scales
    • Short time scales → second
    • Long(er) time scales → seconds, minutes up to hours
  • Examples: daily commute, public transport

• What we need: data rates at medium time scales
  • Data rate → correct, but not highly accurate
  • Time scales → tens of seconds, minutes
Data Rate Prediction

• Data rate prediction (≈ anticipation)
  • Avoid confusion with channel prediction!
• Slotted time model
  • Time slot = duration of HLS segment (e.g. 10 seconds)
  • Perfect prediction (unlimited horizon)
    • Data rate for all (future) time slots known
    • Evaluate anticipated data rates once (“static”)
      → Used in this paper/presentation
  • Uncertain prediction (limited horizon, prediction errors)
    • Data rate with errors known for a limited number of future time slots
    • Reevaluate anticipated data rates in each time slot (“dynamic”)
      → Ongoing work

• Is it feasible?
  • Strong evidence in existing research
  • 1st KuVS Workshop on Anticipatory Networks
Example Scenario

- Daily commute for UPB people
  - Bielefeld to Paderborn
  - Nice rural countryside
    - Boring
    - Partially bad/no mobile phone reception (at least for data)
      - Watch favourite TV series!? 

- **Anticipate** low and high data rate phases
  - Buffer more segments in advance for low data rate phases
    - Avoid buffer underruns
  - Do not greedily buffer too many segments and waste resources
    - Avoid over-buffering
Download Scheduling

• Additional aspect: video quality vs. data rate?
  • Video quality level → file size → required data rate for download
  • Intuitive example:
    • File sizes for different qualities: 3 MB=low, 5 MB=medium, 10 MB=high
    • 1.2 MB/s data rate for 1 time slot of 10 seconds (=12 MB/time slot)
      • 1 high (+2 MB unused)
      • 2 medium (+2 MB unused)
      • 1 medium + 2 low (+1 MB unused)
      • 4 low

• Download schedule
  • When is each segment downloaded in which quality?

• How to compute a download schedule?
• How to implement scheduling in a real system?
MIQCP

- Mixed integer quadratically constrained problem
  - Quality selection broadens solution space
  - When & which quality results in a multiplication of decision variables
- Optimal $\rightarrow$ reference implementation
- Notable running times
  - Acceptable for evaluation
  - Not feasible for real world usage
- Assumes perfect prediction $\rightarrow$ “global” optimum

Fill Algorithm

- Iterative algorithm with backtracking
  - As long as data rate is high
    → Schedule segments in maximum quality for available data rate
  - Encounter low data rate phase
    → Change schedule for previous time slots

- Short running time
- Packing heuristic for quality selection
- Assumes perfect prediction
  - Direct competition to MIQCP

- Details and pseudocode in the paper!
Simulation

- State of the art algorithms in HLS players
  - Greedily fill buffer of fixed size with segments
  - BufferFirst $\rightarrow$ prefer to fill buffer over higher quality
  - QualityFirst $\rightarrow$ prefer high quality over full buffer
  $\rightarrow$ Represents current behavior of VLC

- Scenario
  - 44 base stations in a line
  - Users move from one end to the other as a group
  - Randomly remove base stations to create gaps with no data rate
  - LTE radio model
Simulation Results

- **Observations**
  - Anticipatory scheduling eliminates playback interruptions
  - Fill eliminates lateness at the price of significantly lower quality
  - Buffer usage illustrates how strategies work
    - Greedy → always buffers
    - MIQCP → buffers whenever necessary
    - Fill → buffers late (at the expense of quality)
Real-World Implementation

- Extend HLS protocol
  - Specify buffer size
  - Specify refresh interval
  - Backwards compatible!

- Intercept playlist downloads with transparent proxy
  - Arbitrary content providers
  - Can be implemented by the network operator or external service provider

- Modified version of VLC
Testbed

- 2 smartphones and 2 tablets with Android
- 4 WiFi access points
- Content server
- Management over SSH and ADB (Android Debug Bridge)
Testbed Measurements

- Observations
  - Quality and buffer usage show low difference
  - Higher lateness in the testbed
Testbed Measurements

- Observations
  - Quality and buffer usage show low difference
  - Higher lateness in the testbed
- Real world = real hardware, real software
  - VLC player with real network stack
  - Slotted time vs. continuous time and rounding
  → MIQCP and Fill are too “eager”
Conclusion

- **Anticipatory Download Scheduling for Wireless Video Streaming**
  - Eliminate playback interruptions
  - Maintain good QoE
  - “SmarterPhones”: our approach works

- Testbed implementation shows usability in real world

- **Future/ongoing work**
  - Model for uncertain predictions (limited horizon, prediction errors)
  - Dynamic, error-aware scheduling algorithm
  - Eventually: extended real-world prototype
Uncertain Predictions and Plan Algorithm

![Graph showing the relationship between the number of removed base stations and various metrics such as average latency, average quality, and average buffer size.]
Conclusion

• Anticipatory Download Scheduling for Wireless Video Streaming
  • Eliminate playback interruptions
  • Maintain good QoE
  → “SmarterPhones”: our approach works

• Testbed implementation shows usability in real world

• Future/ongoing work
  • Model for uncertain predictions (limited horizon, prediction errors)
  • Dynamic, error-aware scheduling algorithm
  • Eventually: extended real-world prototype