Multimedia Streaming

SIV864: Special Module on Media Processing and Communication

Tarun Mangla

tarunmangla.github.io

Email: tmangla@iitd.ac.in

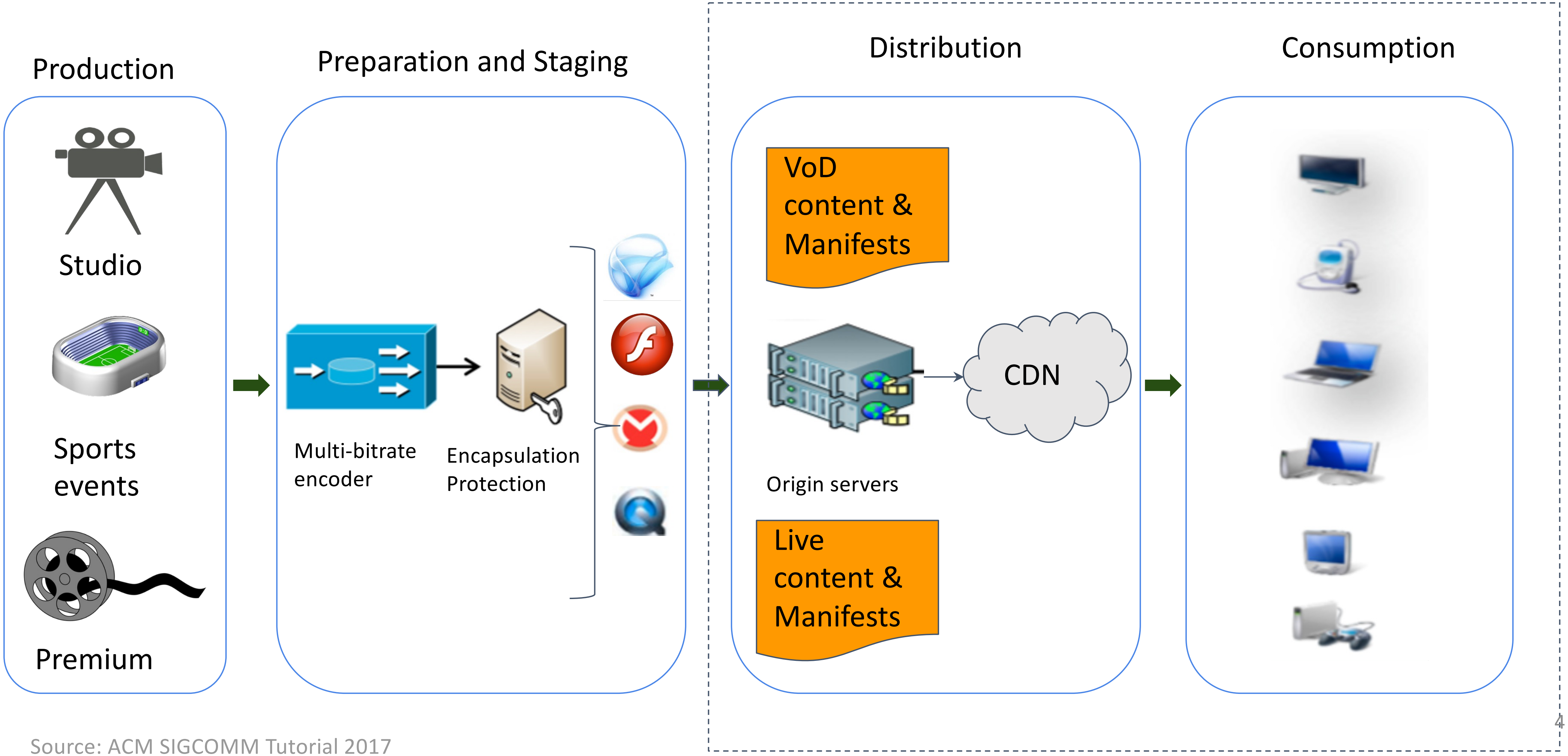
Types of Networked Multimedia Applications

- Stored Video Streaming
- Conversational Video-over-IP
- Streaming Live Video
  - Many techniques similar to stored video streaming

Types of Networked Multimedia Applications

- Stored Video Streaming
  - Streaming: playout begins within few seconds after receiving video
  - Interactivity: pause, repositioning, fast-forwarding
  - Continuous playout: avoid freezing or skipping of frames
- Conversational Video-over-IP
  - Real-time encoding
  - Delay sensitive
  - Loss tolerant

Streaming Stored Media: 10,000 FT View



Source: ACM SIGCOMM Tutorial 2017

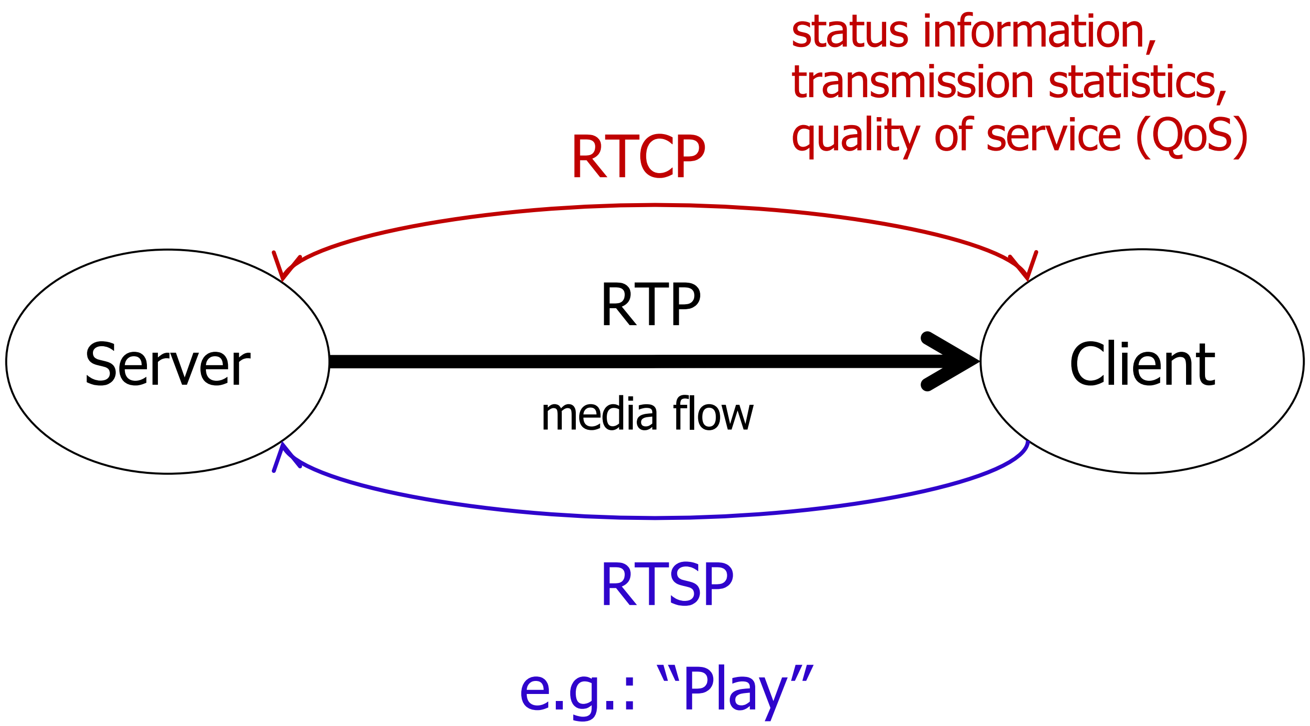
Distributing Stored Video

RTP/RTSP/RTCP

- Push-based protocol

Streaming Protocol Suite (1)

- Flow diagram: RTP, RTCP, RTSP



Distributing Stored Video

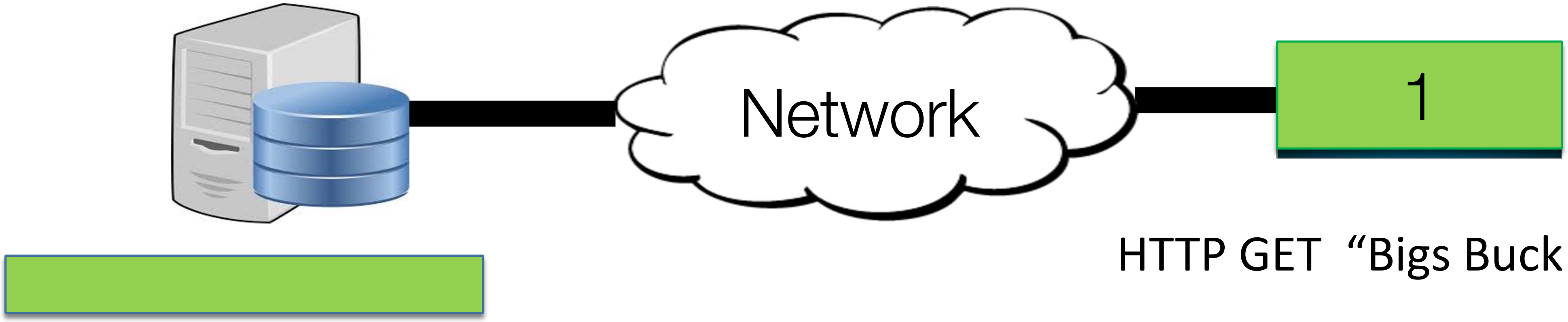
<u>RTP/RTSP/RTCP</u>

- Push-based protocol
- RTP for data transmission, RTSP for playback control (pause, rewind, play etc.), RTCP for synchronization and control
- Cons: Specialized hardware (stateful server), Firewall issues

<u>HTTP</u>

- Pull-based protocol
- Prefetching content to mitigate network variations
- Pros:
  - Stateless server: Re-use existing web infrastructure
  - TCP provides congestion control and reliability

HTTP Progressive Download



HTTP GET "Bigs Buck Bunny"

Bigs Buck Bunny

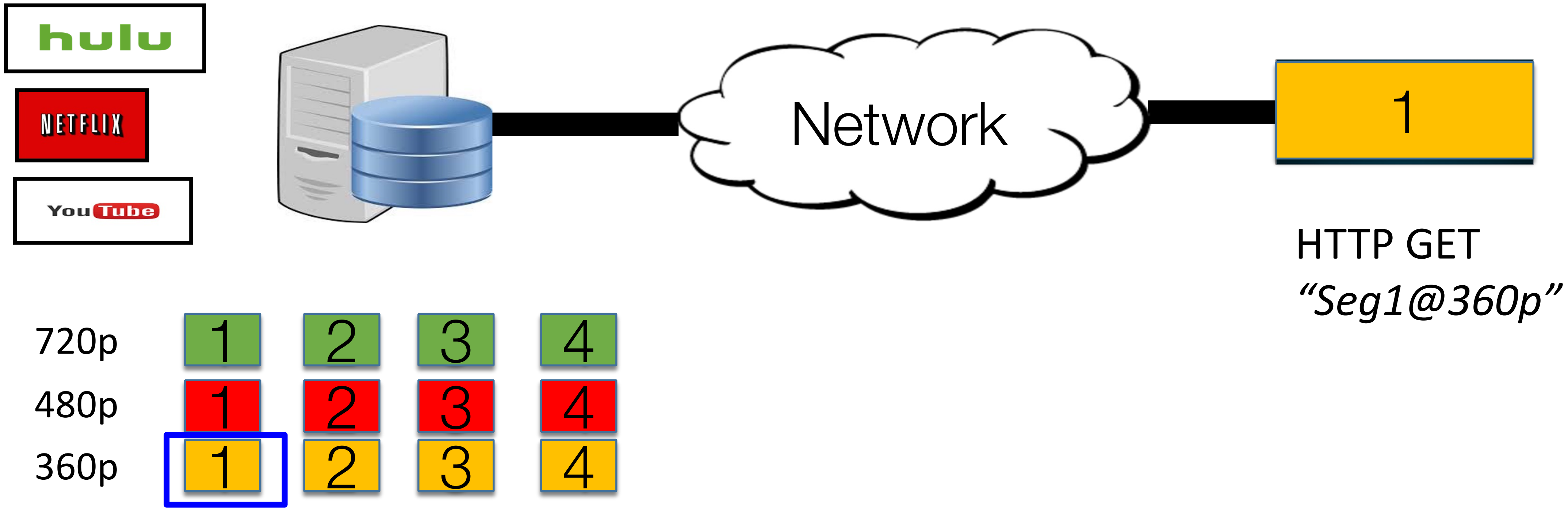
How to implement interactivity?

- Chunking the file
- HTTP byte-range requests

What resolution should we stream the content? [Client and Network diversity]

- Real-time encoding [not efficient]
- Pre-encode multiple versions

HTTP Adaptive Streaming (HAS)



Supports diverse clients and network conditions

Existing HAS standards

Popular standards

- MPEG-DASH
  - http://reference.dashif.org/dash.js/v2.6.2/samples/dash-if-reference-player/index.html

- HLS implemented by Apple
  - http://video-dev.github.io/hls.js/demo/

Legacy/Almost dead standards

- Microsoft Smooth Streaming
- Adobe Flash

DEMO

[DASHIF Implementation]

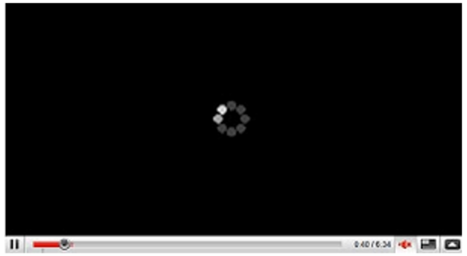
Key Functionalities of the HAS Player

- Fetches manifest file
- Uses OS-provided HTTP stack to download video segments
- Decrypts video content
- Performs bitrate adaptation
- Logging for analytics

Goal of Bitrate Adaptation

- Bitrate adaptation aims to optimize the **Quality of Experience** (QoE)
- QoE is subjective; challenging to infer at scale
- Objective metrics are used

Minimize Re-buffering



Maximize average bitrate

Minimize bitrate switches

Minimize startup latency

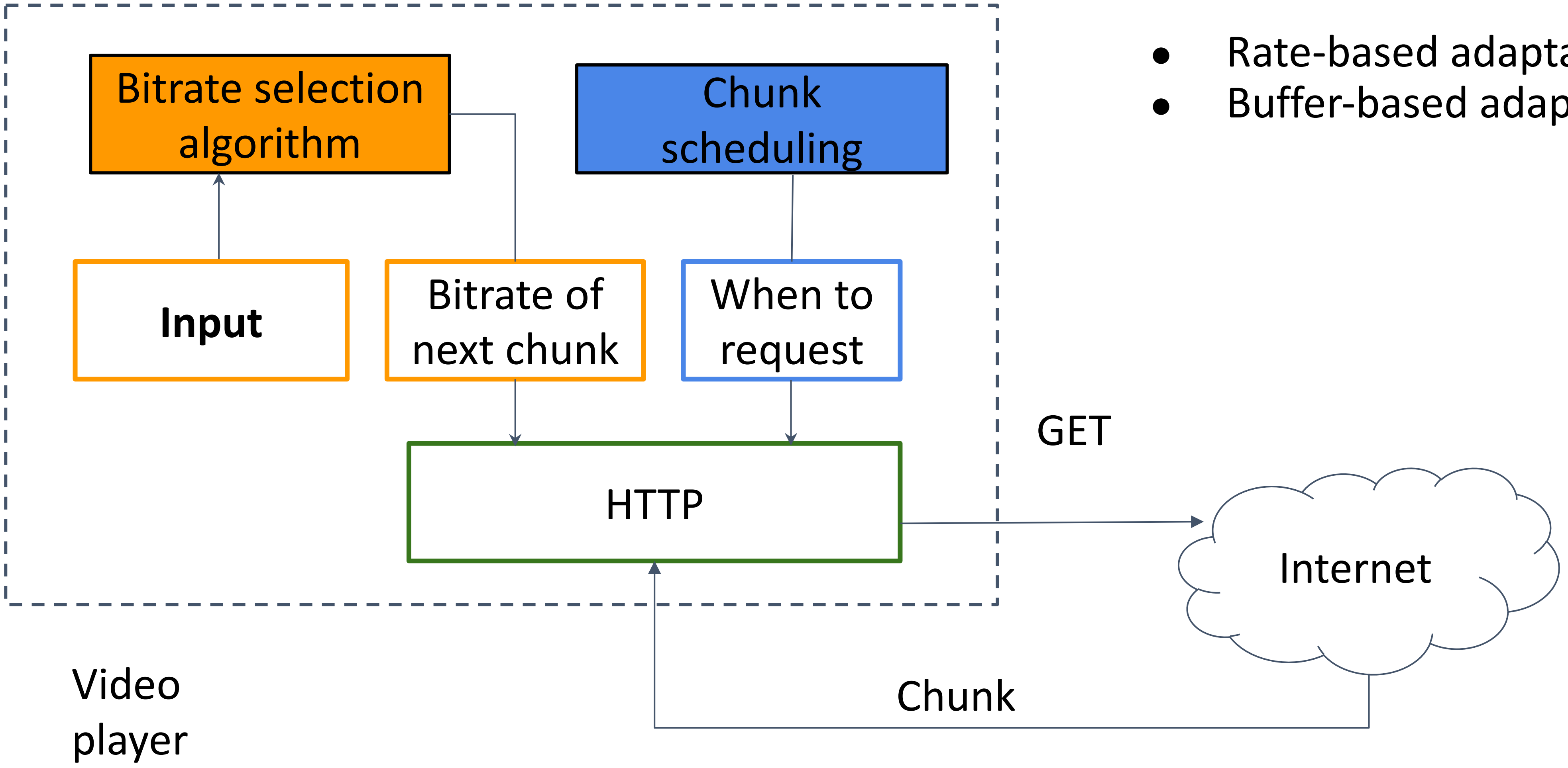
What Factors Bitrate Depends On?

- Rendering capability (screen resolution, player window size)
- Computation capability (CPU load)
- Network throughput

Static

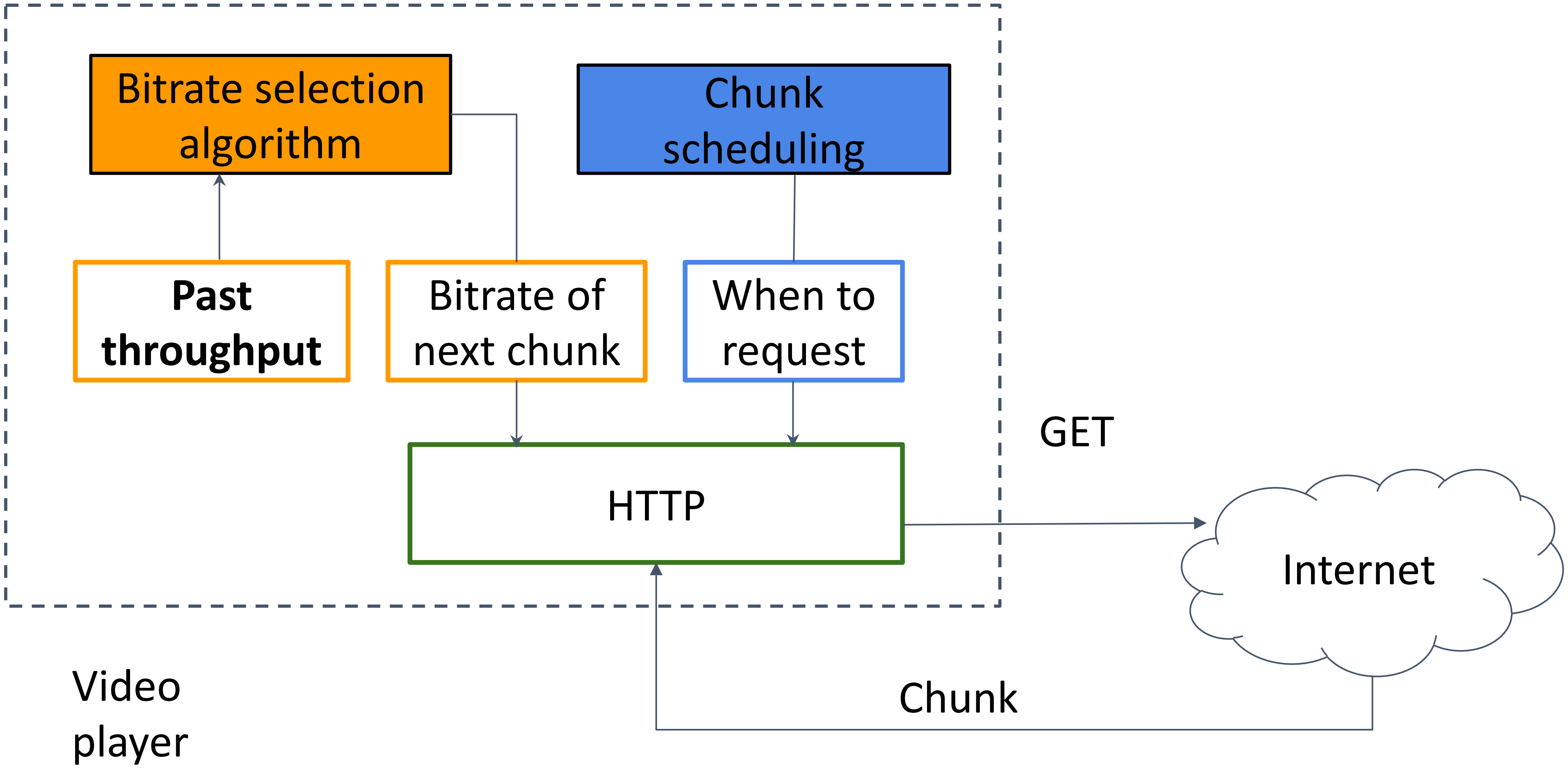
Minimal impact

Abstract player model

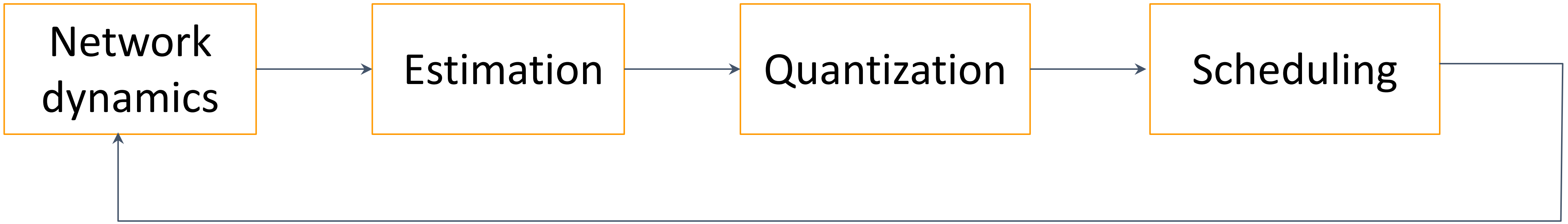


- Rate-based adaptation
- Buffer-based adaptation

Rate-based Bitrate Adaptation



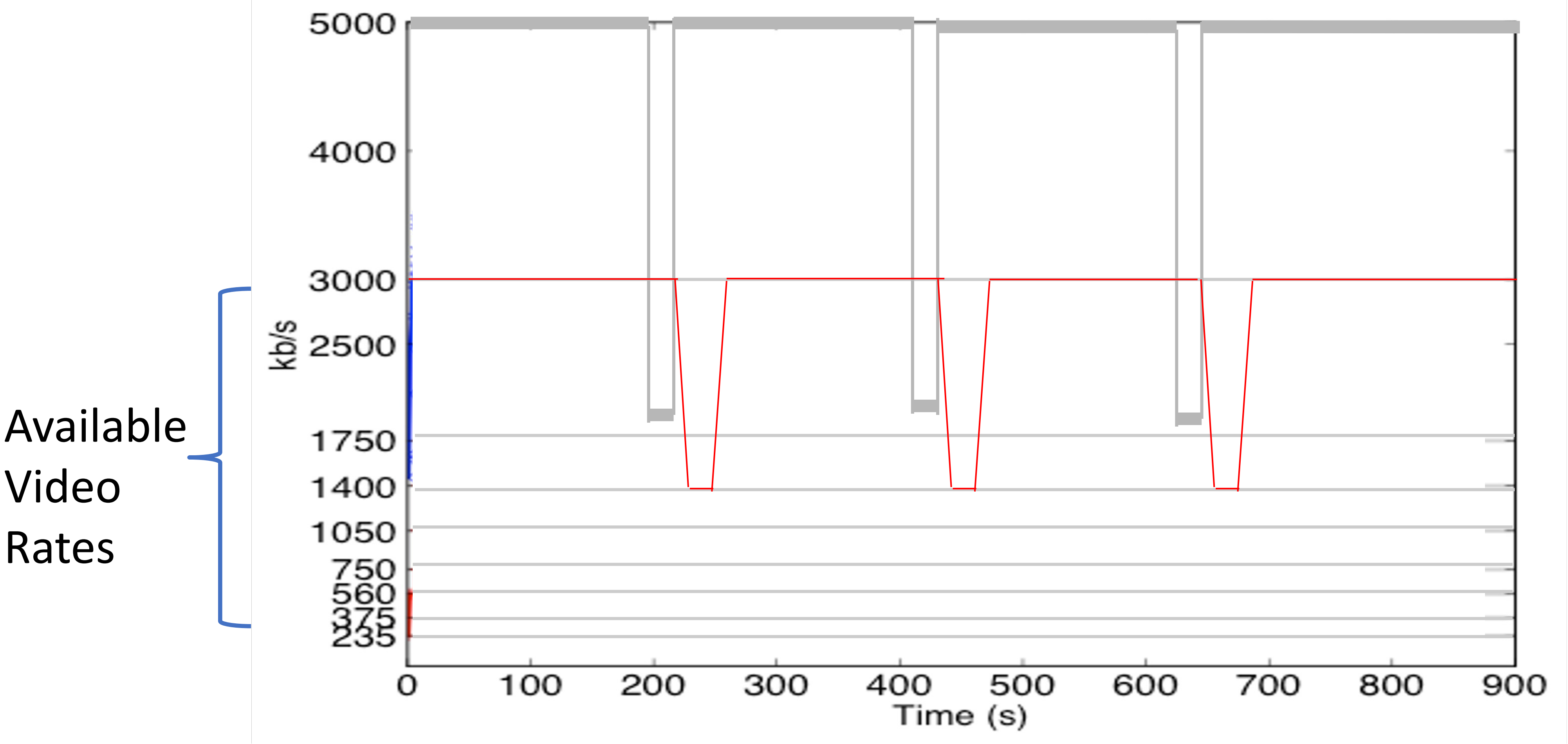
Simple Rate-based Adaptation



1. **Estimation**: Use last chunk throughput (***T_{n-1}***) to estimate future network bandwidth
2. **Quantization**: Continuous throughput is mapped to discrete bitrate
$$\max_{r \in R} (1 + \alpha) \times r \leq T_{n-1}$$
3. **Scheduling**: Specifies how much time to wait before requesting the next chunk

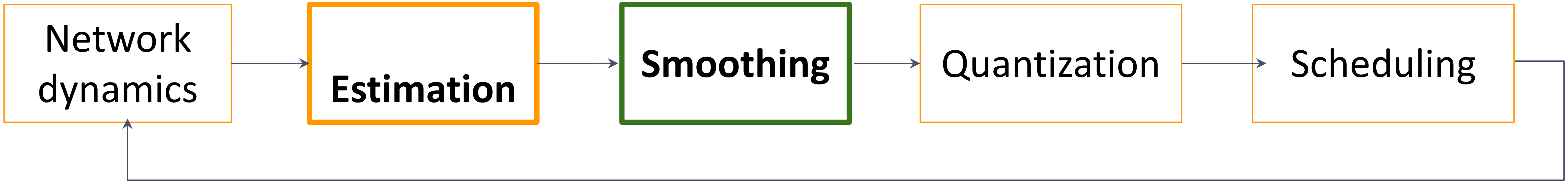
What happens under network throughput variations?

The Case of Bitrate Oscillations



Using only the last-chunk throughput leads to *unnecessary* bitrate switches

Modified Rate-based Adaptation



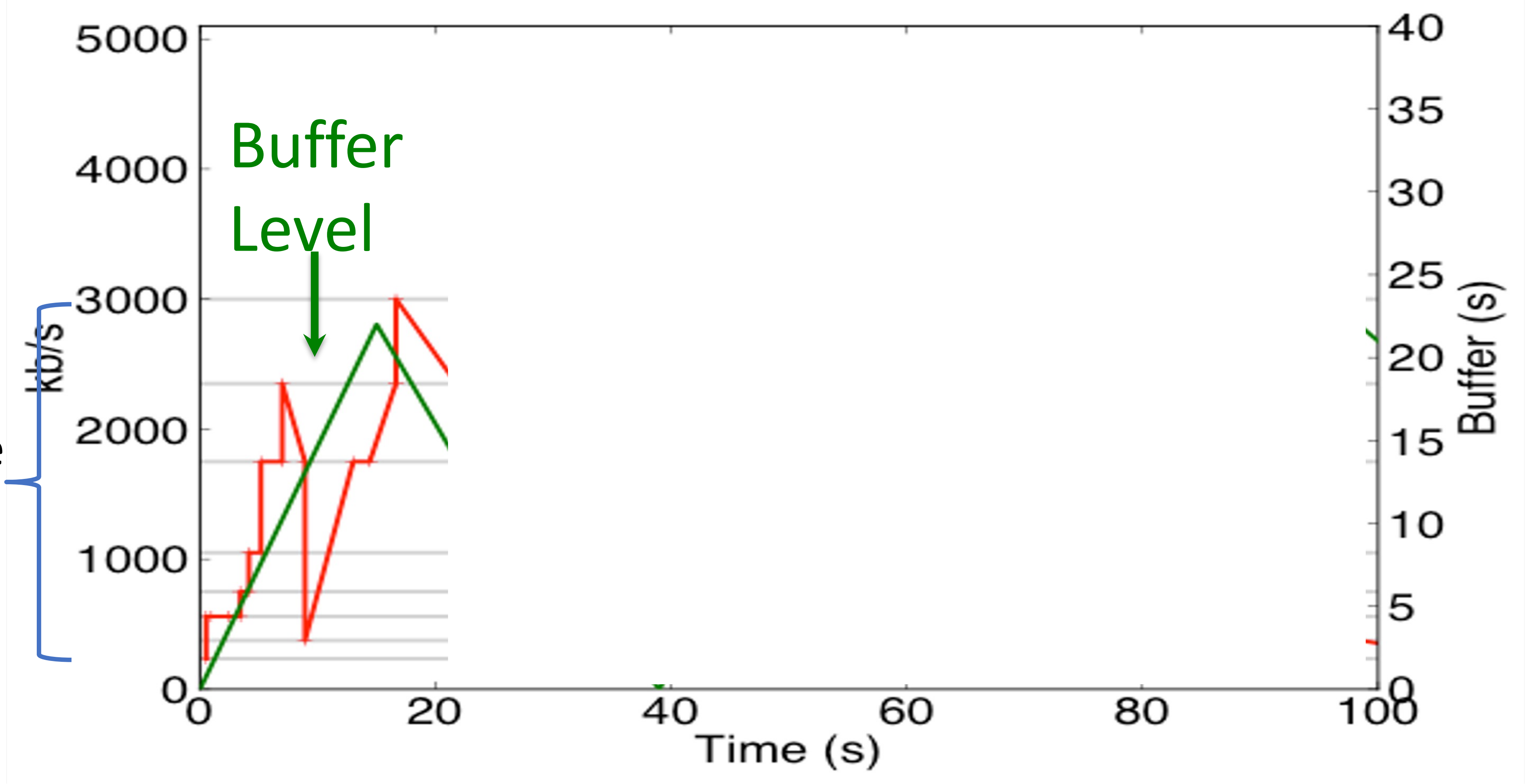
1. **Estimation: Take into account historical values, not just the last chunk throughput**
2. **Smoothing: Apply a smoothing filter such as average, harmonic mean or EWMA**
3. Quantization: Continuous throughput is mapped to discrete bitrate
4. Scheduling: Specifies how much time to wait before requesting the next chunk

Are we done?

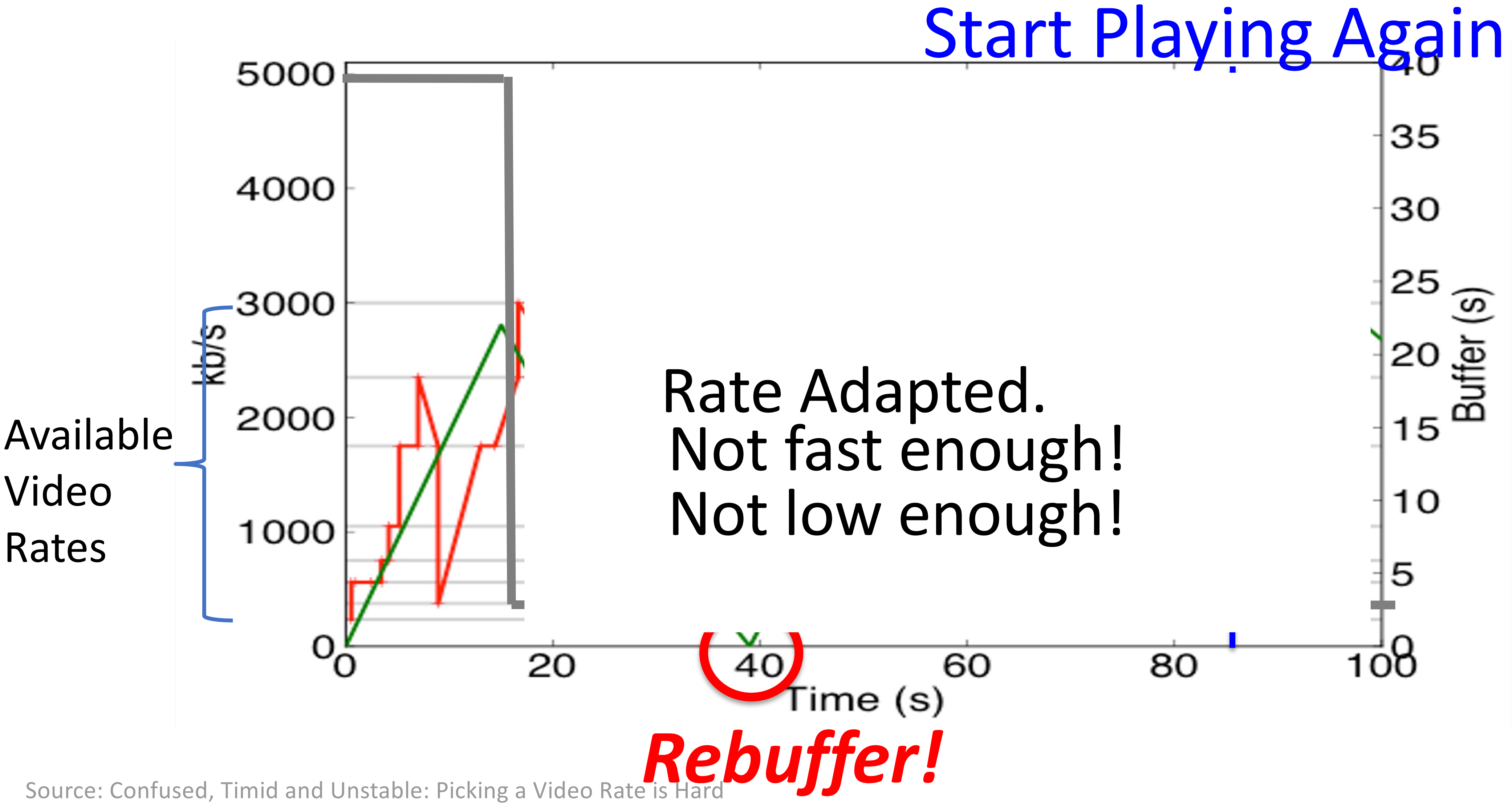
Cons: Reactive adaptation, Background traffic

What happened when bandwidth decreased?

Available Video Rates

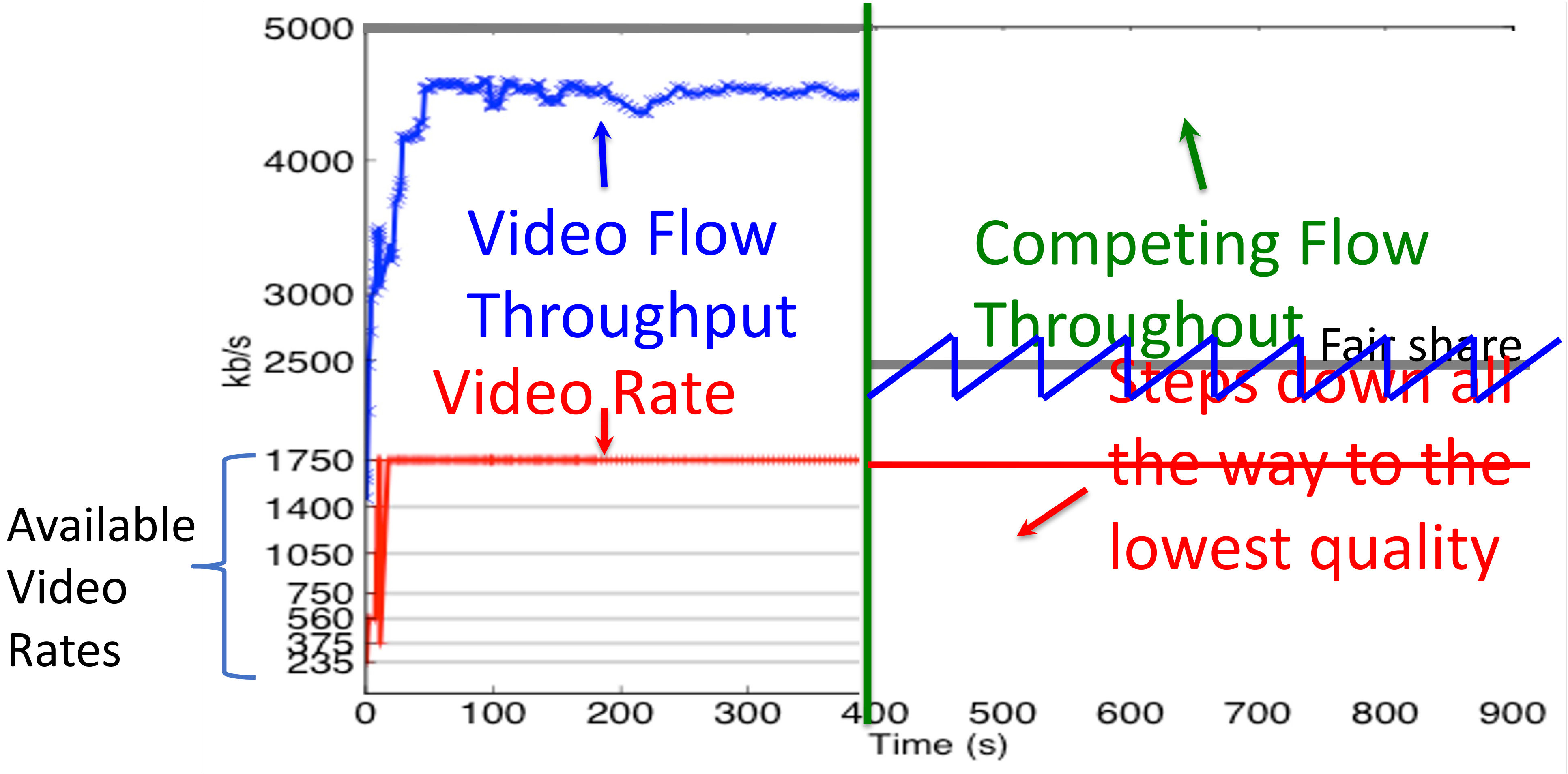


What happened? – Cont.



Source: Confused, Timid and Unstable: Picking a Video Rate is Hard

The case of competing flow



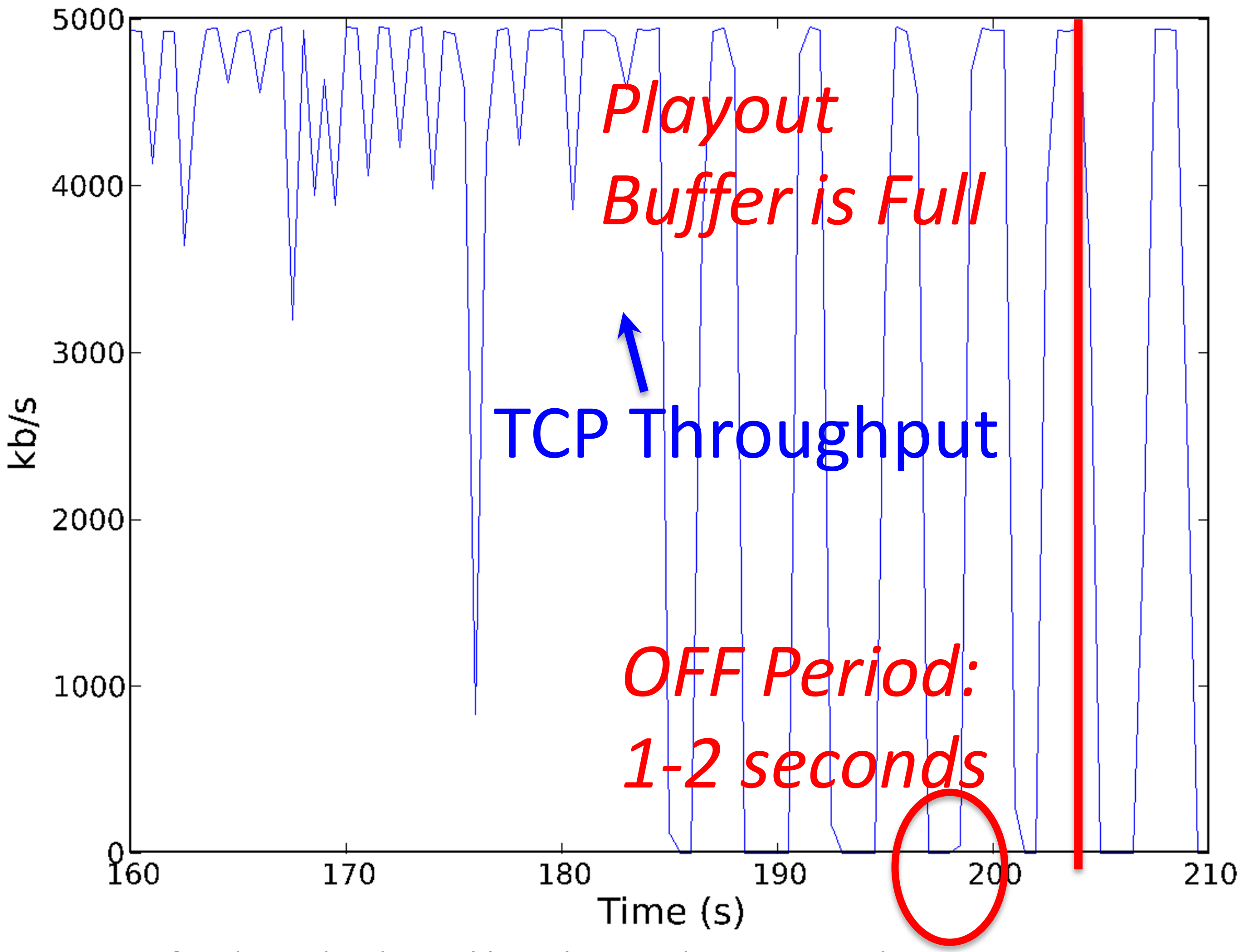
The Problem

Video client ends up with much less throughput than its fair share

It picks a video rate that is much too low

Why?

TCP Throughput of the Video Flow

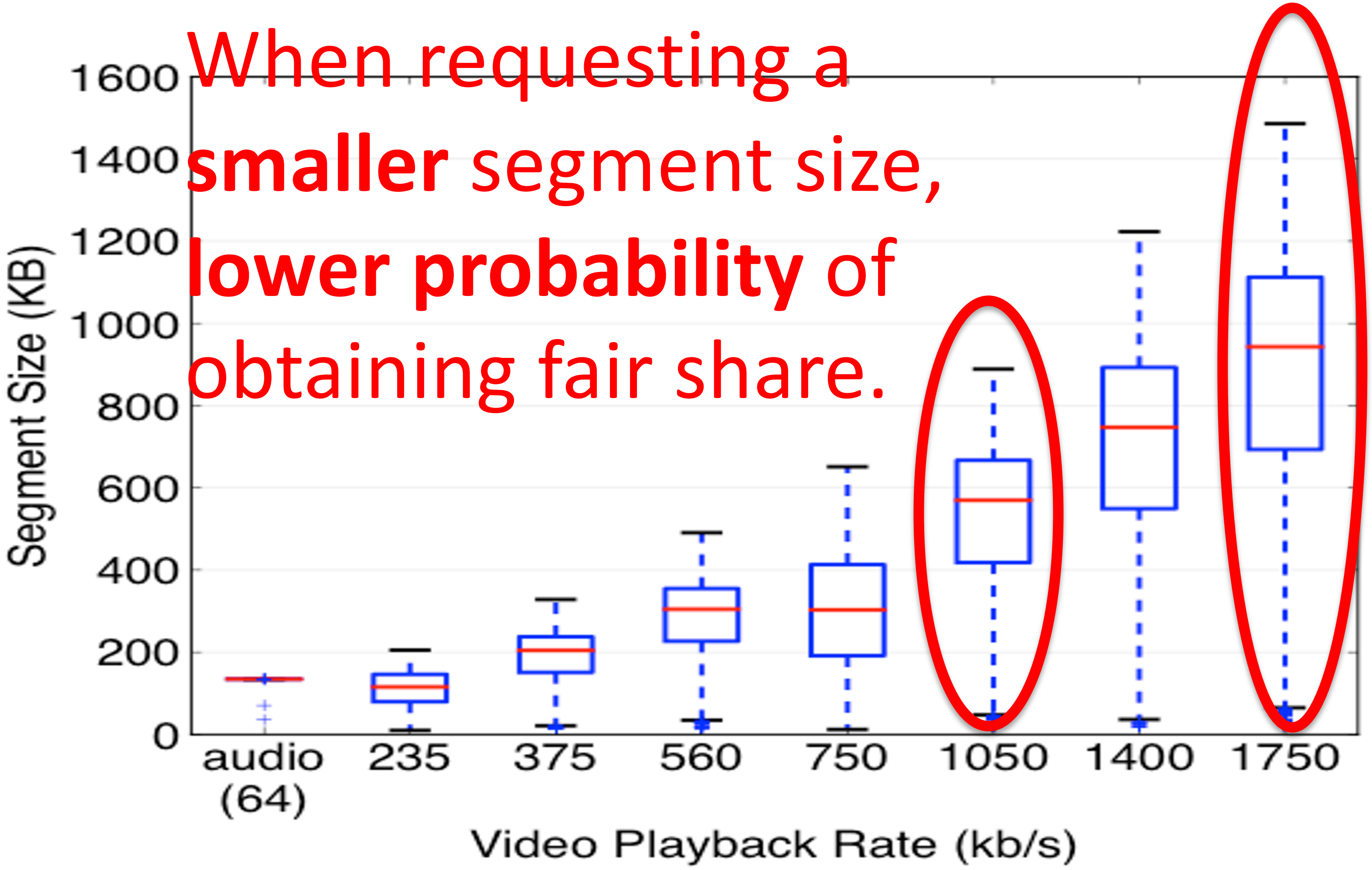


- TCP sender resets its congestion window during OFF period
- Throughput will be affected especially with a competing flow
- Experience packet loss during slow start
- 50% of the segments get < 1.8Mb/s

Source: Confused, Timid and Unstable: Picking a Video Rate is Hard

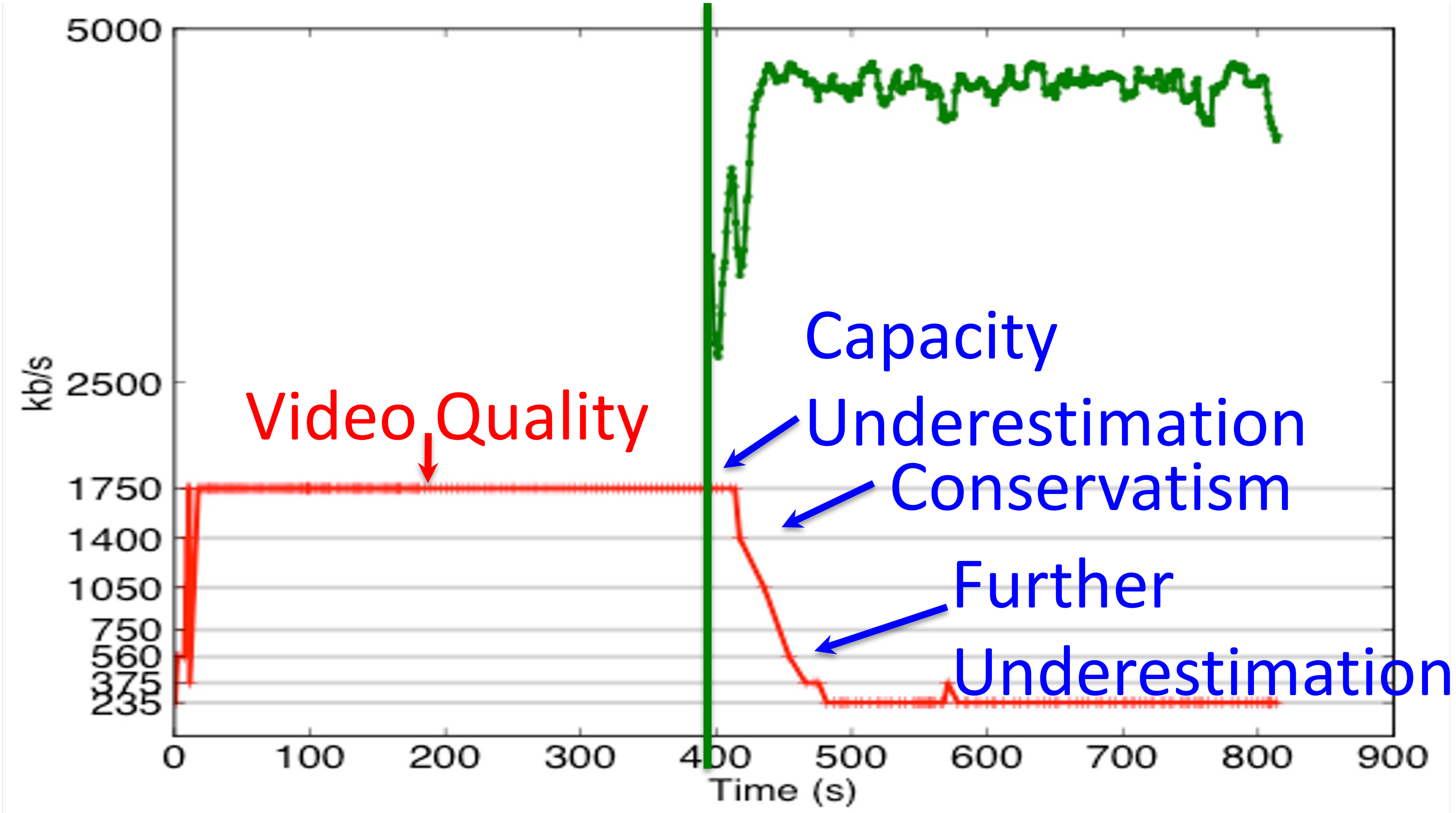
Smaller Segment Size for Lower Video Rate

When requesting a **smaller** segment size, **lower probability** of obtaining fair share.



Source: Confused, Timid and Unstable: Picking a Video Rate is Hard

The Complete Story



Being conservative can trigger a vicious cycle!

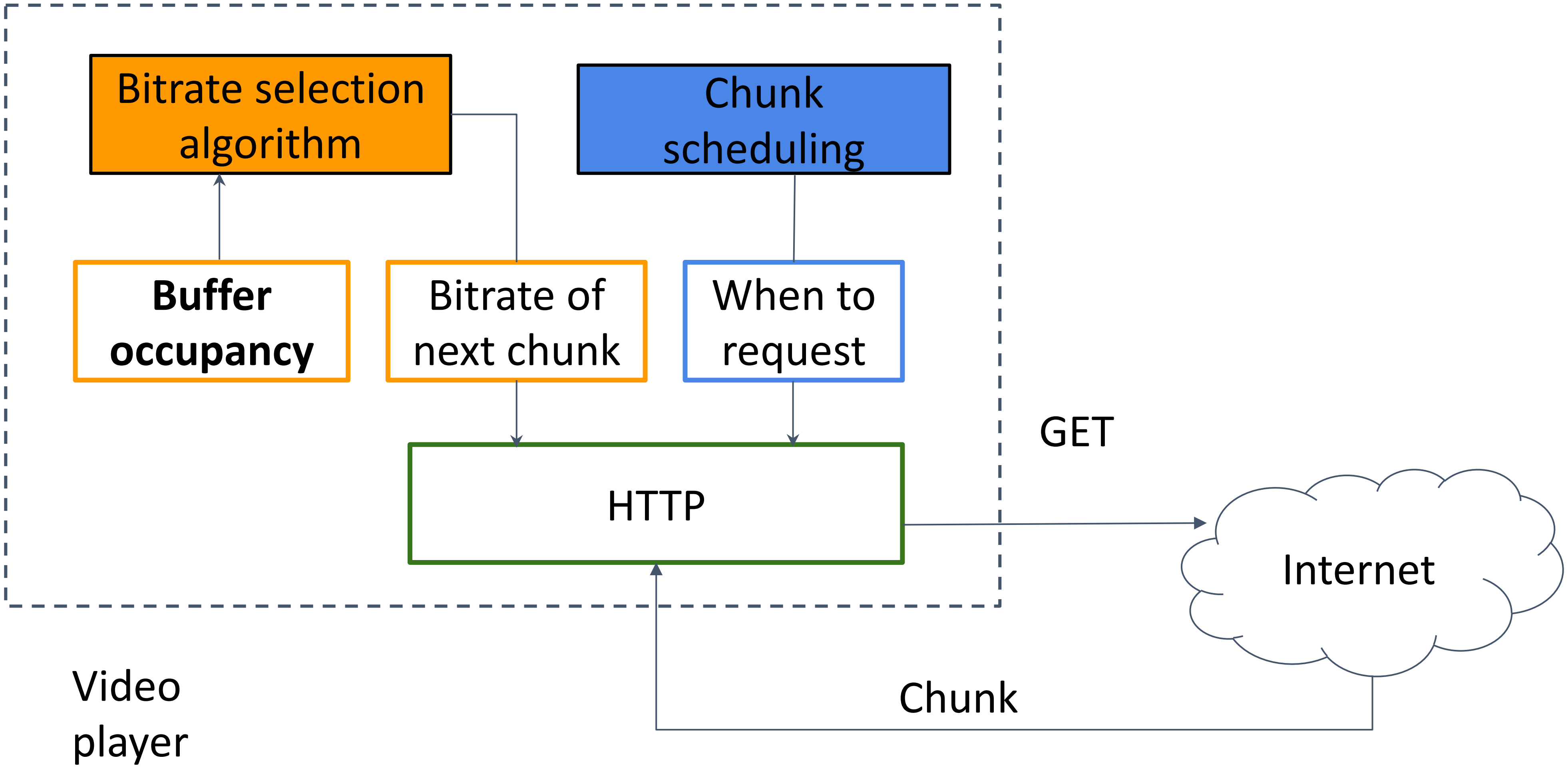
Source: Confused, Timid and Unstable: Picking a Video Rate is Hard

Problem With Rate-based Adaptation

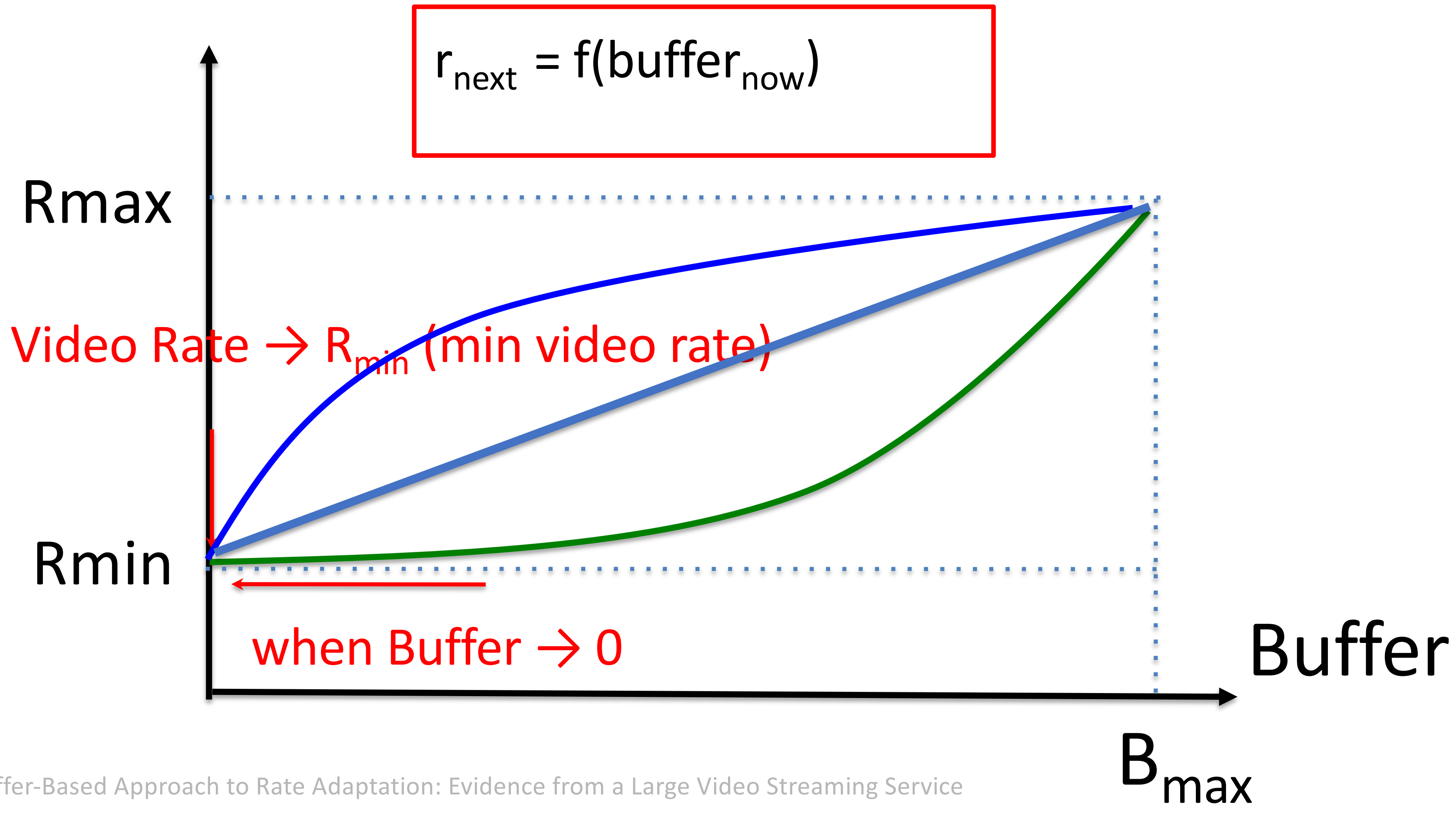
- Pick rate based on capacity estimation
  - The actual capacity is unknown and varies
  - The *reactive* estimation usually does not match the actual capacity
- The same algorithm can both **under-estimate** and **over-estimate** the capacity

Source: Confused, Timid and Unstable: Picking a Video Rate is Hard

Buffer-based Bitrate Adaptation



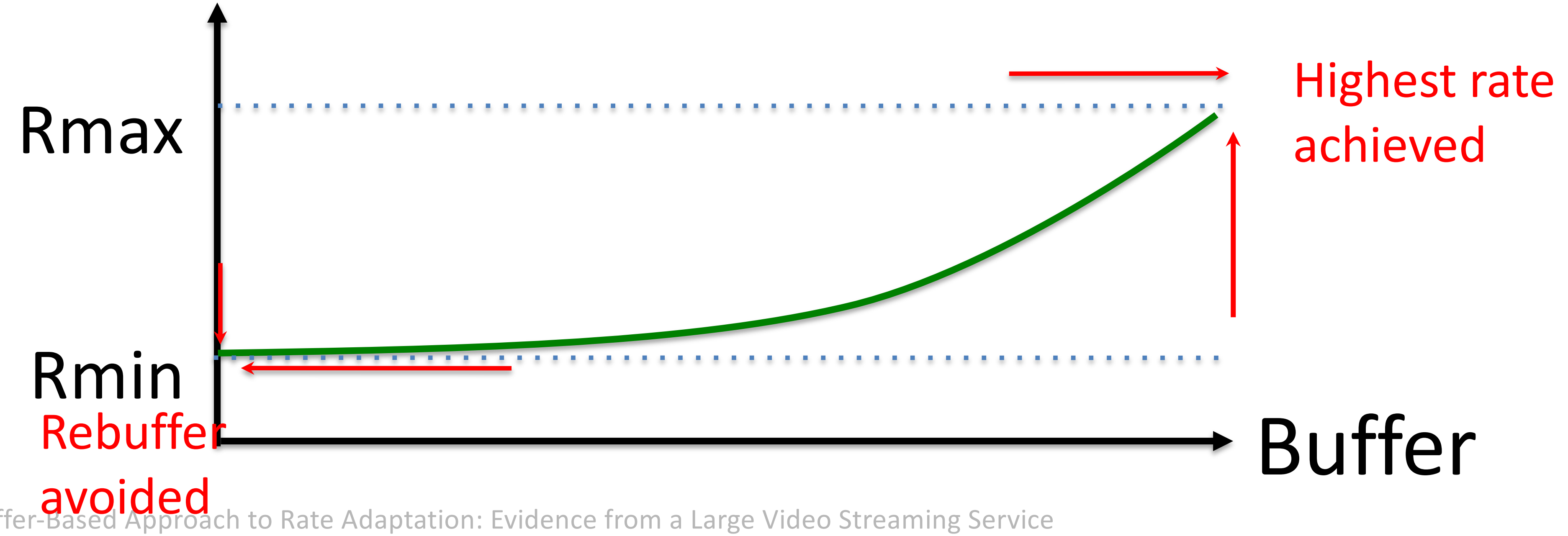
Buffer-based adaptation: Algorithm Sketch



Source: A Buffer-Based Approach to Rate Adaptation: Evidence from a Large Video Streaming Service

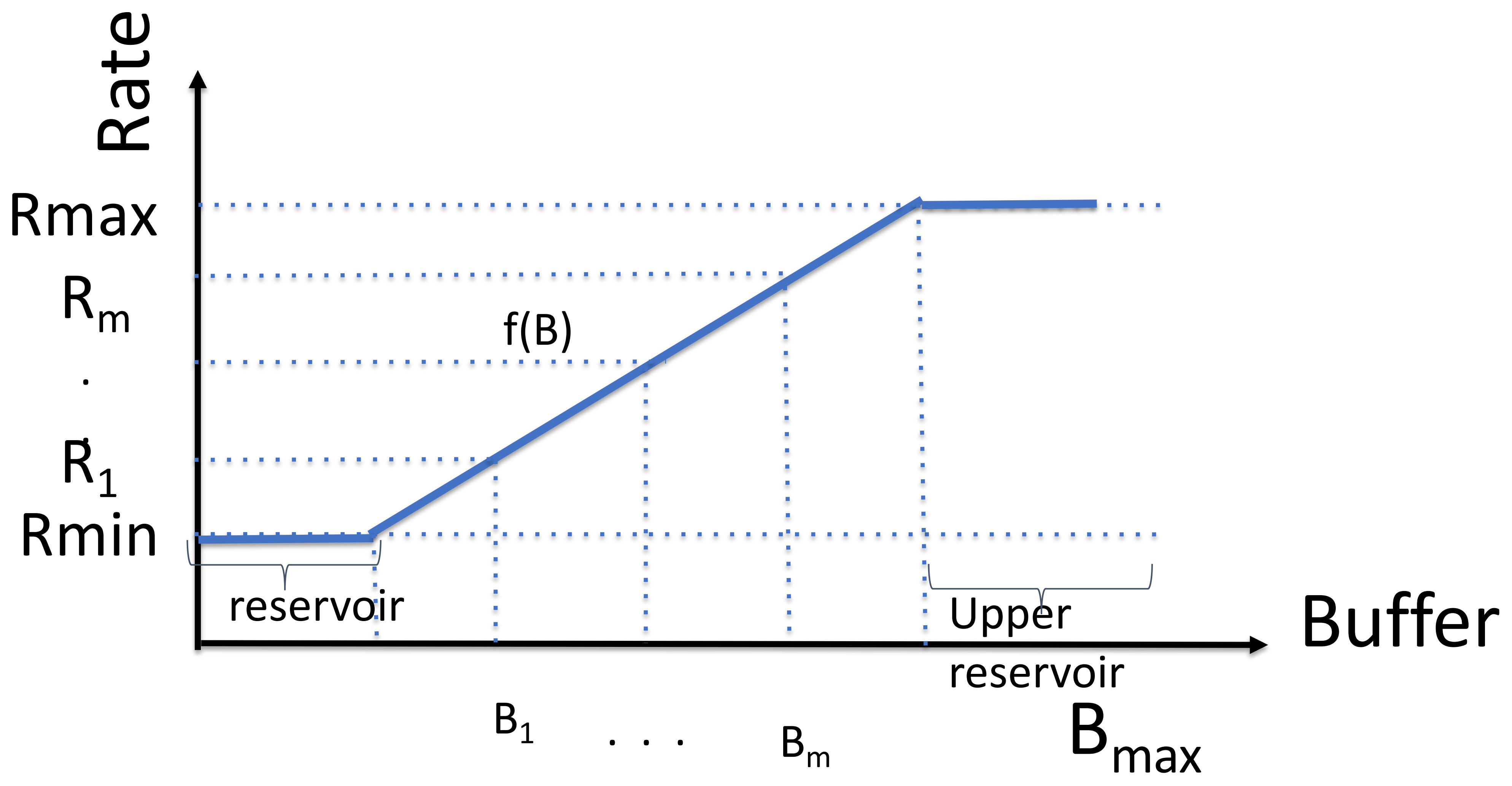
Advantages of buffer-based adaptation

- Utilize the full capacity of the link
  - Avoid on-off behavior as long as the video quality is less than maximum
  - Request the highest video rate before the buffer is full
- Avoid "unnecessary" re-buffering
  - Reduce the bitrate as the buffer occupancy decreases



Source: A Buffer-Based Approach to Rate Adaptation: Evidence from a Large Video Streaming Service

Buffer-based adaptation: Algorithm



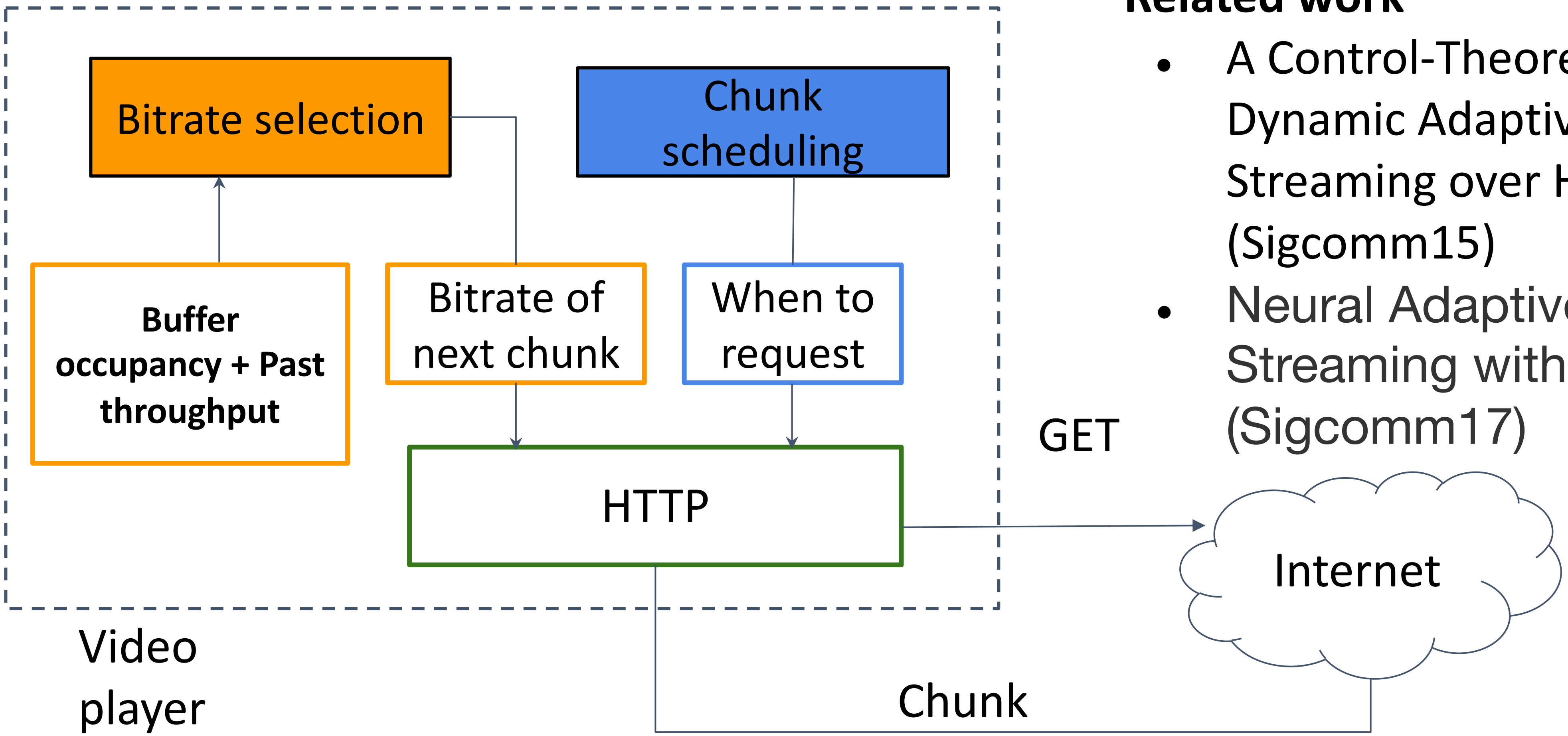
Problems with buffer-based adaptation

- Low video quality during the ramp-up phase
- Unnecessary bitrate oscillations
- Requires large-buffer which might not be available for live content

What to do now?

What to do now?

For best results, use both buffer occupancy and past throughput



Related work

- A Control-Theoretic Approach for Dynamic Adaptive Video Streaming over HTTP (Sigcomm15)
- Neural Adaptive Video Streaming with Pensieve (Sigcomm17)

Summary

- HTTP-based adaptive streaming (HAS) used for delivering Internet video
- Bitrate adaptation is important to ensure a high Quality of Experience (QoE)
- Various bitrate adaptation algorithms have been proposed
  - Rate-based: Rely on past observed throughput
  - Buffer-based: Rely on current buffer occupancy
  - Other methods: Control theory approach, machine learning
- Open problems: Bitrate adaptation, encoding, storage, server selection …