Optimal Scalable Video Multiplexing in Mobile Broadcast Networks

Farid M. Tabrizi, Cheng-Hsin Hsu,

Mohamed Hefeeda, and Joseph G. Peters

Network Systems Lab, Simon Fraser University, Canada

Deutsche Telekom Lab, USA

SFU

Outline

- Motivations
- Mobile Video Broadcast Networks
- Problem Statement and Formulation
- Our Solution
- Evaluation Results
- Conclusions

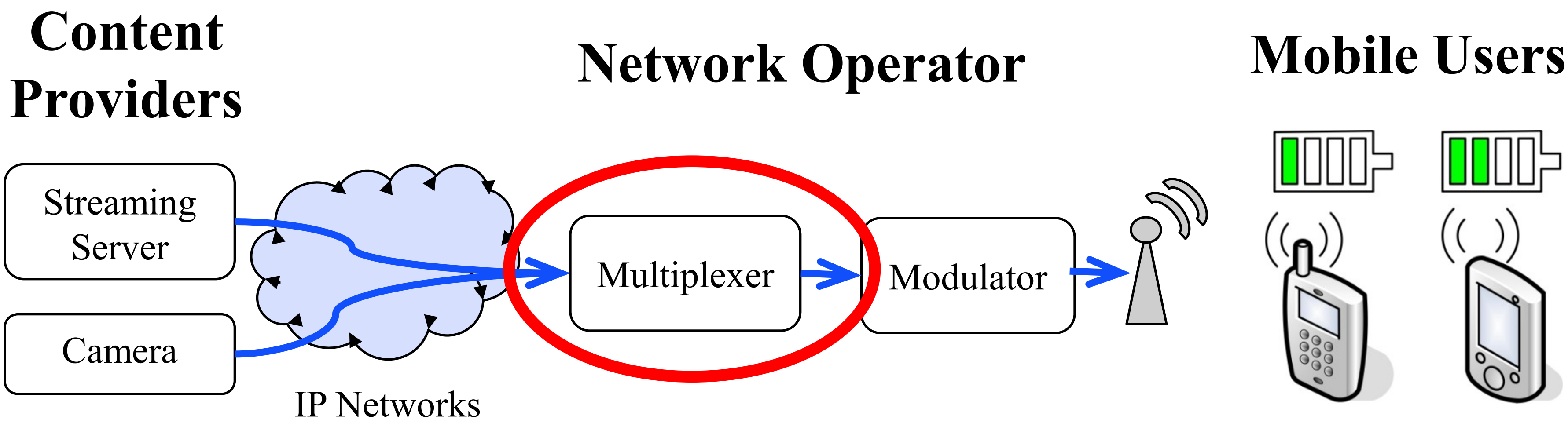
SFU

Motivations

- **Mobile videos are getting increasingly popular**
- **However, delivering mobile videos over unicast channels of cellular networks is inefficient**
  - ***Analysis*** **predicted that 3G cellular networks would collapse with only 40% mobile phone users watching 8-minute video each day [Liang et al. PTC'08]**
  - **AT&T is phasing out their unlimited data plans**
- **More efficient delivery method is needed**
- **We study broadcast networks that support multicast/broadcast for higher spectrum efficiency**

SFU

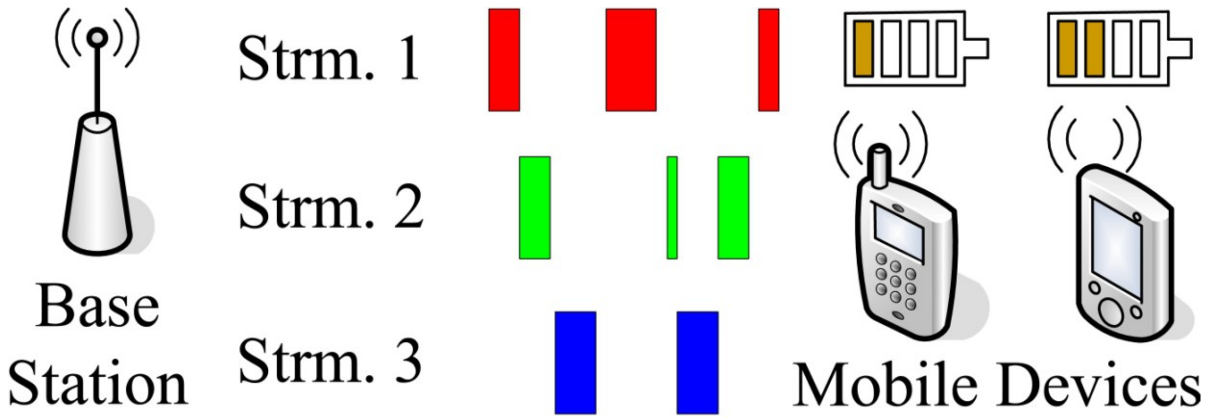
Mobile Broadcast Networks



- Content providers create videos for recorded and live programs
- Network operator multiplexes multiple videos into a broadcast stream
- Mobile users receive the broadcast stream

We studied the design of multiplexers

Challenges



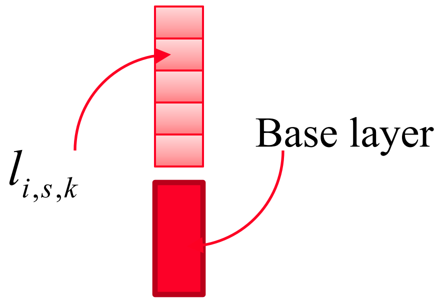
- **Designing multiplexer is not easy**
  - Small buffer sizes of mobile receivers
  - Energy constraints for mobile receivers
  - Variability in the bitrates of video streams
- **Goal: a real-time scheduling algorithm to**
  - Maximize number of broadcast streams in the network
  - Minimize energy consumption on mobile receivers
  - Maximize the overall video quality

Medium-Grained Scalable Streams

- Modern H.264/SVC codec supports two types of quality scalability: coarse-grained scalability (CGS) and medium-grained scalability (MGS)
- CGS enables layer-level adaptation
  - Switching between frames is only possible at I-frames
  - The choice among different bitrates is limited by no. layers
- MGS allows packet-level adaptation
  - Switching at any frame
  - Many more bitrates are possible
- We leverage on MGS coded streams

Problem Statement

- **Problem: Broadcasting S MGS video streams from a base station to a large number of mobile receivers over a shared wireless medium**
- **Notations:**
  - **There are S video streams**
  - **Each frame video stream s has a base layer and Q_s MGS layers**
  - **Each video stream has I frames**
  - $l_{i,s,k}$ **Indicates the size of layer k of frame i of stream s**
  - **Each stream is coded at F frame-per-second**



Frame i of stream s

Formulation

Pri: $$\max \ \sigma = \frac{\sum_{s=1}^{S} \sum_{j=1}^{n_s} b_j^s / R}{I/F},$$ ← Goodput, fraction of ontime delivered data

Sec: $$\max \ \gamma = 1 - \frac{\sum_{s=1}^{S} \sum_{k=1}^{n_s} (T_0 + b_k^s / R)}{I/F} / S,$$ ← Energy saving, fraction of network interface off-time

Sec: $$\max \ \phi = \frac{\sum_{s=1}^{S} \sum_{k=1}^{n_s} \sum_{i=g_k^s}^{h_k^s} \sum_{q=1}^{u_i^s} \lambda_{i,s,q}}{\sum_{k=1}^{n_s} b_k^s},$$ ← Overall video quality in PSNR

s.t. $$[f_k^s, f_k^s + \frac{b_k^s}{R}) \cap [f_k^{\bar{s}}, f_k^{\bar{s}} + \frac{b_k^{\bar{s}}}{R}) = \varnothing,$$ ← No burst overlapping

$$c_k^s + b_k^s - \sum_{f_k^s \leq j/F < f_k^s + b_k^s/R} \sum_{q=0}^{u_i^s} l_{j,s,q} \leq B,$$ ← No buffer overflow

$$c_k^s \geq 0,$$ ← No buffer underflow

$$b_k^s \geq \sum_{i=g_s^k}^{h_s^k} \sum_{q=0}^{u_i^s} l_{i,s,q},$$ ← Bursts are large enough to accommodate selected video packets

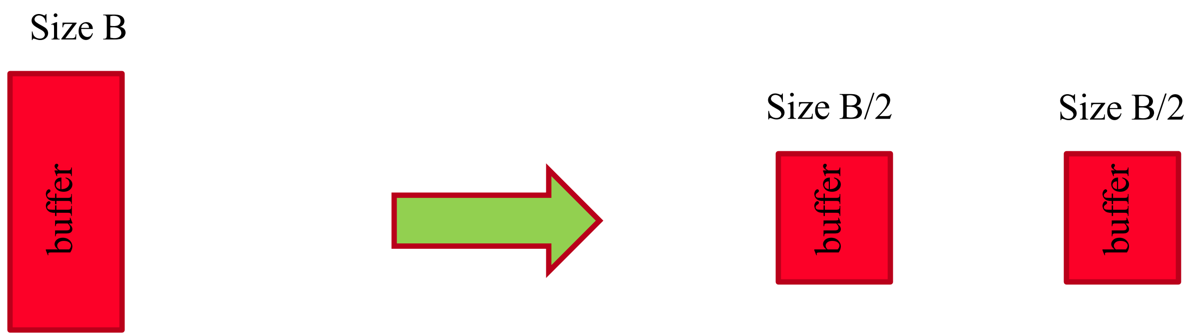
SFU

Problem Solution

- **Split receiver buffer of size *B* to two buffers of size *B/2***
- **For each video stream, we assign time windows**
- **At each time window of each video stream, one buffer is being drained while the other buffer is being filled**
- **Earliest-deadline-first scheduling in each window**
- **When the draining buffer is empty, we switch the buffers**
- **If due to bandwidth limitations a complete video cannot be sent, we drop MGS layers in a rate-distortion optimized manner and schedule a burst for the empty buffer**

SFU

Double Buffering Technique



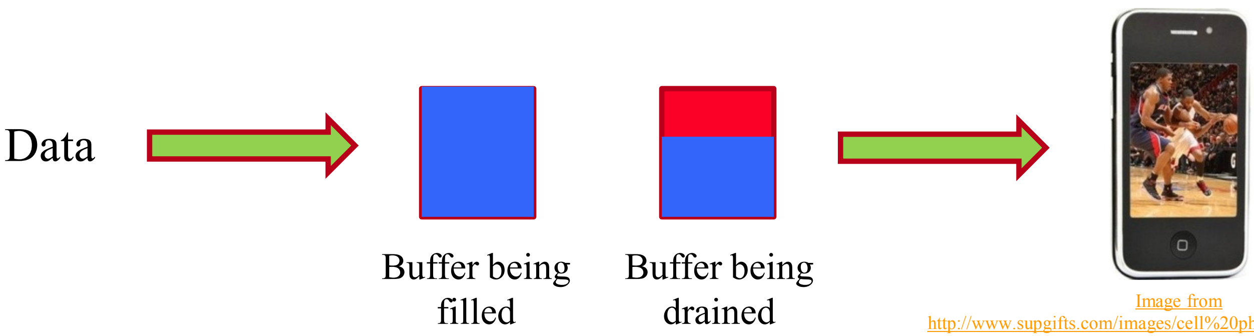


Image from
http://www.supgifts.com/images/cell%20phones/Cellphone652.jpg

SFU

Evaluation Setup

- **Use a MobileTV testbed developed in our lab**
  - **The base station: a Linux box with RF signal modulator implementing the physical layer of mobile broadcast protocol**
  - **Indoor antenna to transmit DVB-H compliant signals**

- **Settings**
  - **We set the modulator to use 16-QAM (Quadrature Amplitude Modulation)**
  - **10MHz radio channel**
  - **Transition overhead time** T_o=100 ms

SFU

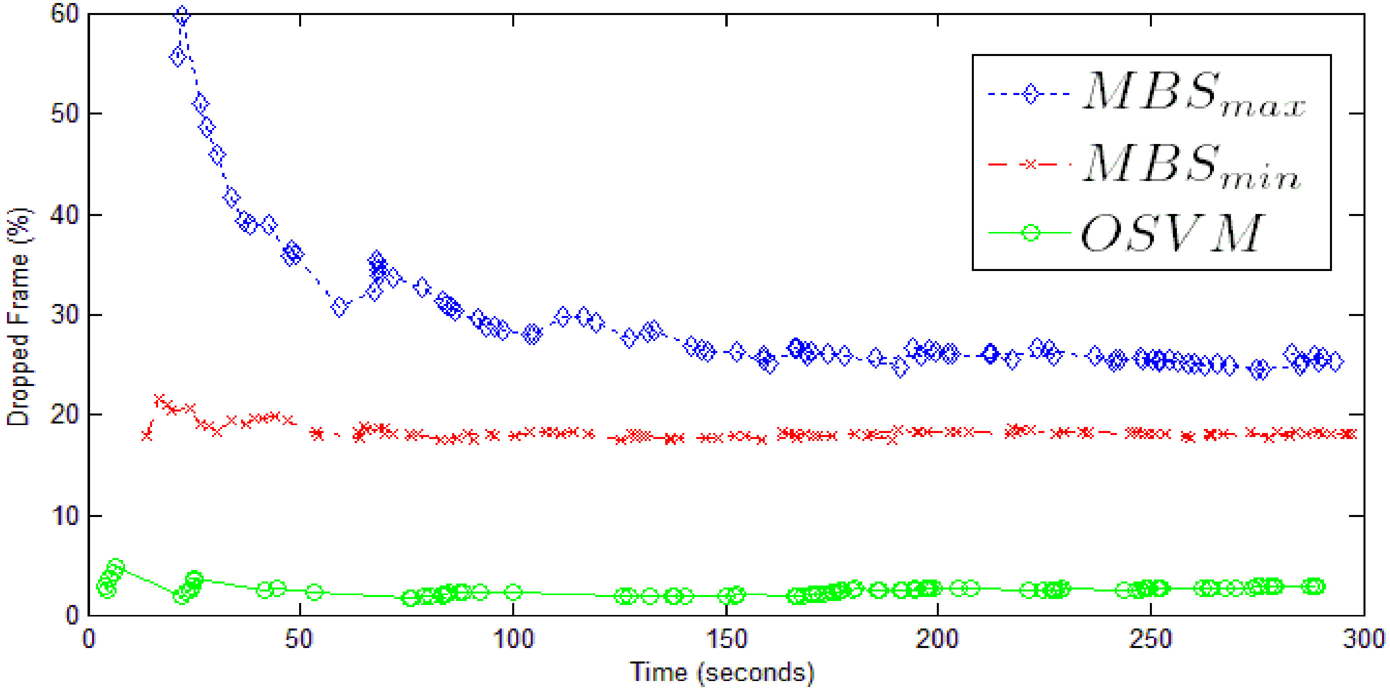
Evaluation Setup (cont.)

- **Video streams**
  - 10 video streams of different categories of: sport, TV game show, documentary, talk show and have very different visual characteristics
  - Bitrates ranging from 250 to 768 kbps
  - We created video streams with different MGS layers and the trace file for each stream using JSVM
- **Comparison**
  - We compare our **OSVM** algorithm with MBS (Mobile Broadcast Solution) from Nokia and SMS algorithm [MM'09] which has been previously developed in our Lab

SFU

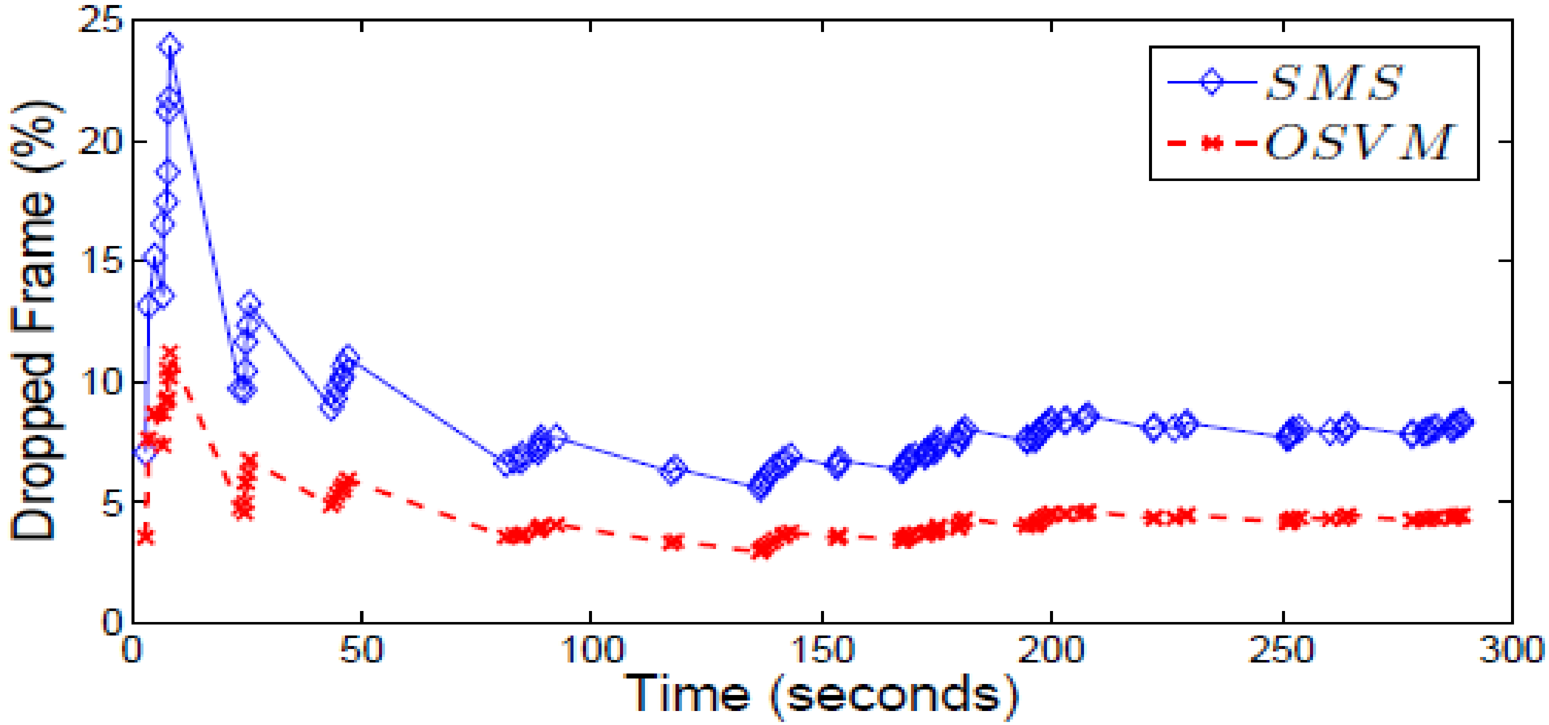
Comparison again Current Base Station

- **We compared our OSVM with MBS algorithm in its best and worst cases (by tuning its parameters)**
- **OSVM algorithm reduces the dropped frame rate from at least 20% to less than 5%**



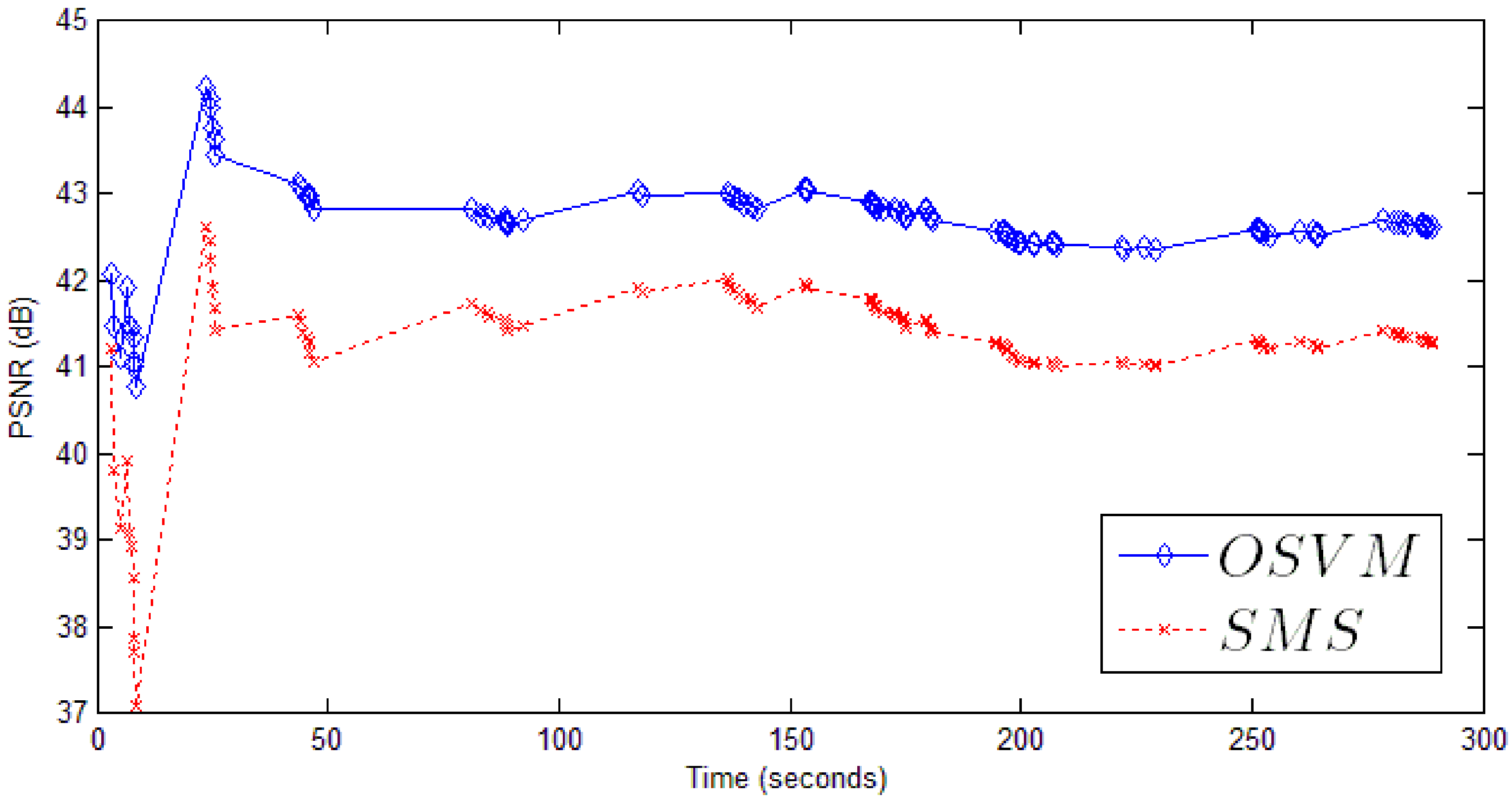
Comparison against Our Prior Work

- **OSVM algorithm results in 46% lower frame drop rate**



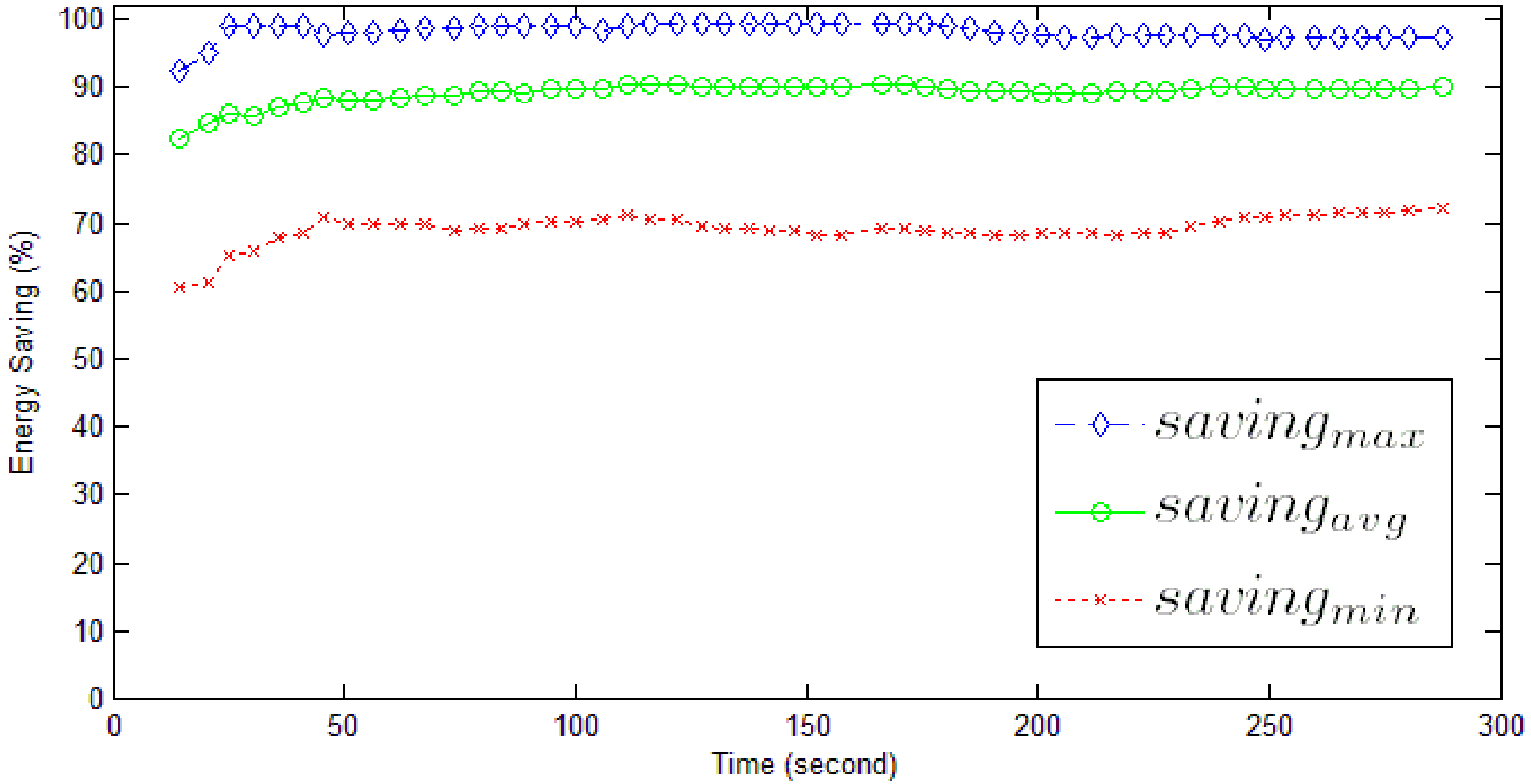
Comparison against Our Previous Work (cont.)

- **OSVM achieve quality improvement of 1.34dB on average**



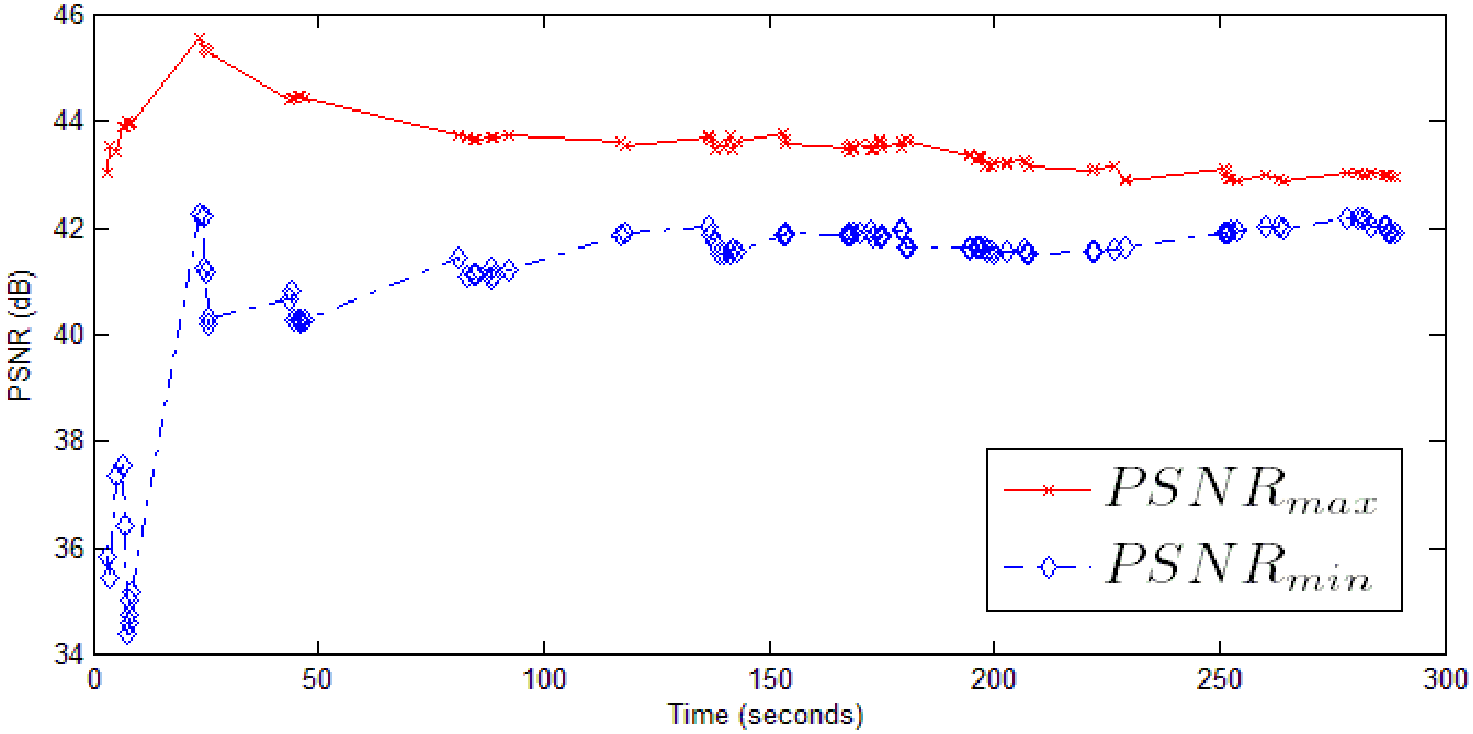
Per-Stream Energy Saving

- **The energy saving resulted from OSVM for all video streams ranges from 70% to 99%**



Per-Stream Video Quality

- The gap between maximum and minimum video quality among all streams is only 1dB



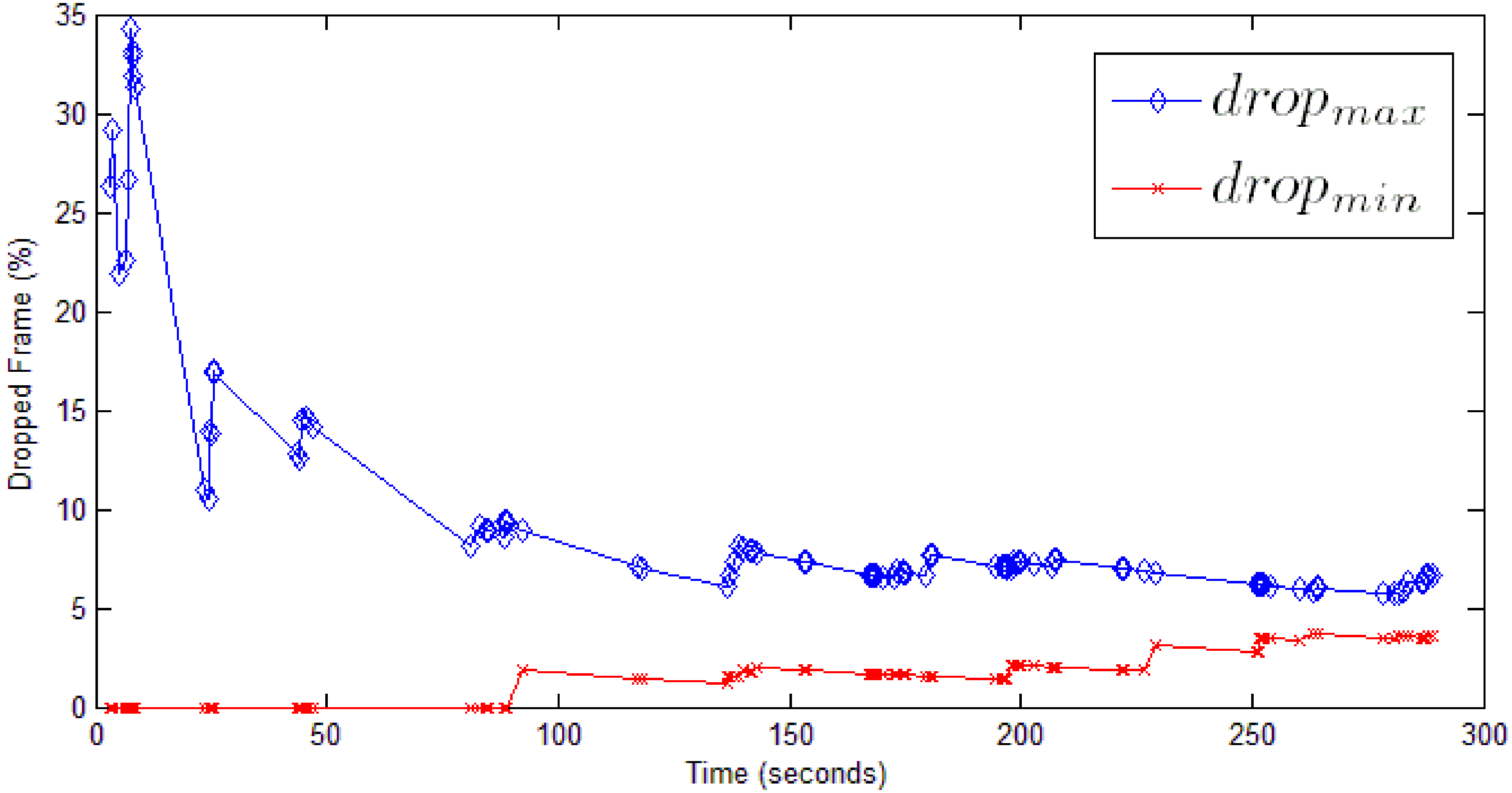
Conclusions

- We studied scalable video broadcast networks
- We formulated a burst scheduling problem to jointly optimize: (i) video quality, (ii) network goodput, and (iii) receiver energy consumption.
- We proposed an efficient algorithm for the problem
- We implement the proposed algorithm in a real mobile TV testbed
- Extensive experimental results indicate that our algorithm outperforms the algorithms used in current base stations and proposed in our previous work [MM'09]

Thank You

Fairness on Frame Drop Rate

- The frame drop rate among all video streams quickly converges to the range of 4.49% to 6.6%



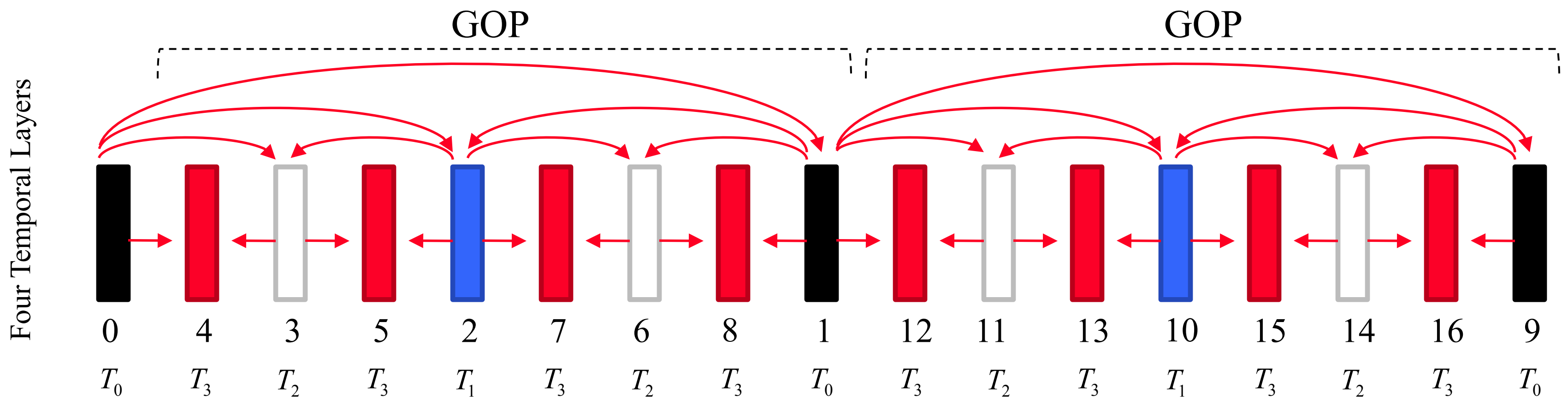
Future Work

- Making the solution adaptive based on the changes in bitrate of video streams
- Considering the effect of larger lookahead window on the performance of multiplexing algorithm
- Using other scalability opportunities like temporal scalability

SFU

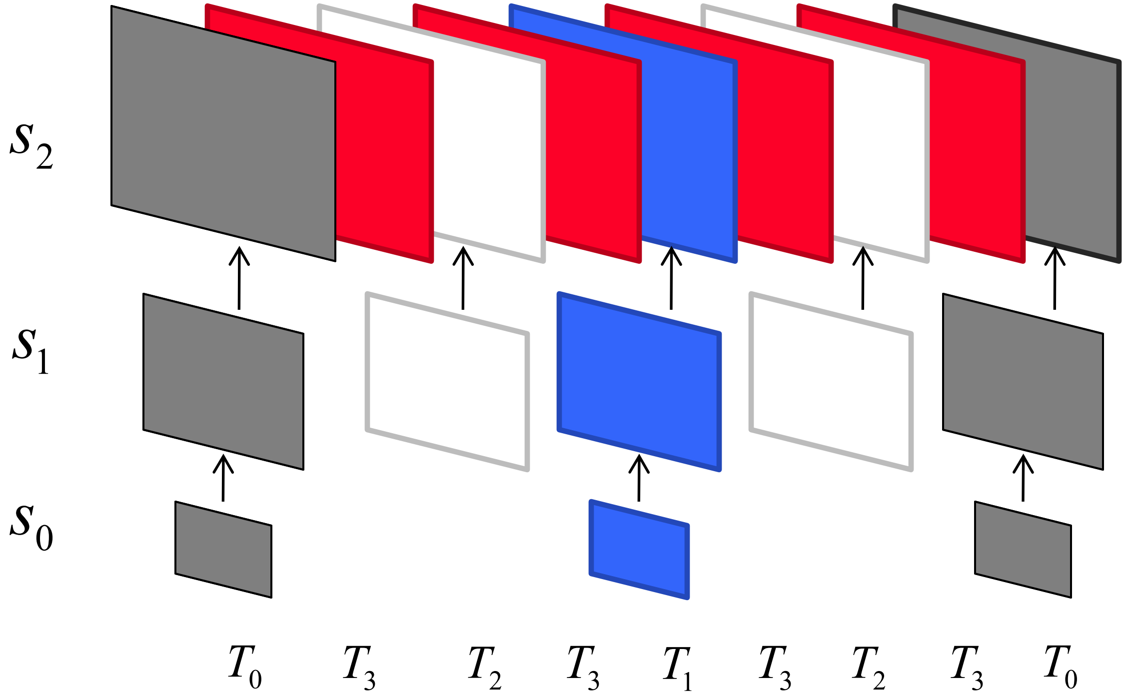
Scalable Video Coding

- **Scalable video coding**
  - Temporal scalability
  - Spatial scalability
  - Quality scalability
- **Temporal scalability**
  - The frames must be encoded in hierarchical prediction structure



Spatial Scalability

- Images with different spatial resolutions
- Each layer in the spatial scalable video stream improves the final image resolution



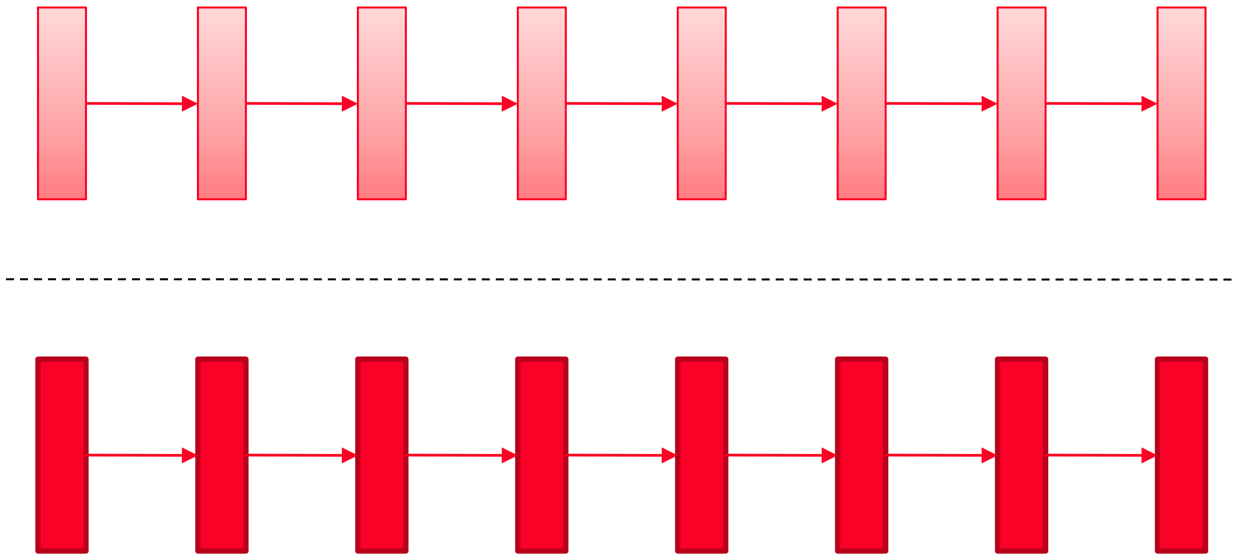
SFU

Quality Scalability

- Quality scalability could be considered as a special case of spatial scalability
- Dividing the video into several quality layers: Coarse Grain Scalability (CGS)
  - In CGS, motion estimation is conducted in each spatial layer separately
    - Switching between frames is only possible at I-frames
    - The choice among different bitrates is limited to the number of layers

SFU

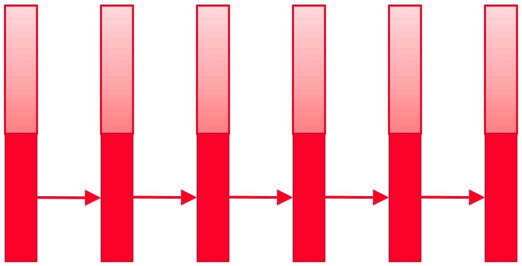
Quality Scalability



Coarse Grain Scalability

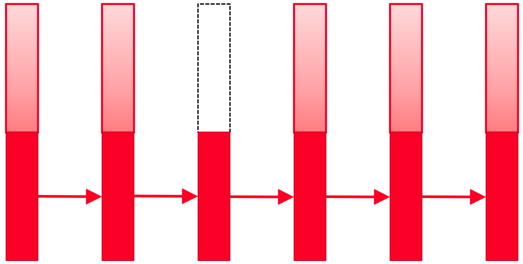
Quality Scalability

- **Alternatives for CGS:**
  - All quality levels in one spatial layer
- **Fine Grain Scalability**
  - Motion compensation is done at the lowest quality level of the reference picture



Quality Scalability

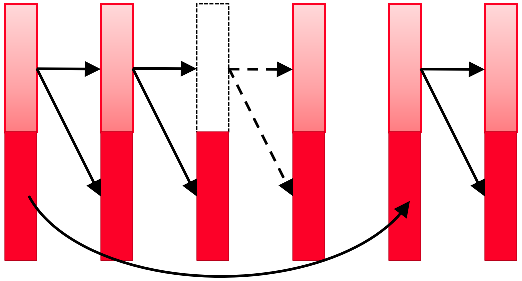
- **FGS advantages:**
  - Encoder and decoder use the same quality level of the reference picture
  - Bitrate scaling could be done at packet level



- **FGS disadvantage:**
  - Coding efficiency

Quality Scalability

- **Medium Grain Scalability**
  - A trade-off between Fine Grain Scalability and Coarse Grain Scalability
  - Keeps drift at an acceptable level
  - Motion prediction done in the enhancement layer with periodic updates at base layer



Definitions

- **Bandwidth utilization**
  - The fraction of video frames received at the decoder before their decoding deadline

$$\sigma = \frac{\sum_{s=1}^{S} \sum_{j=1}^{n_s} b_j^s / R}{I / F}$$

- **Energy saving**
  - The fraction of time the receivers can put their wireless receivers into sleep
  - We use the average energy saving among all video streams

$$\gamma = (\sum_{s=1}^{S} \gamma_s) / S$$

Problem Formulation

- The average quality of all transmitted frames is shown by φ
  - We use peak-signal-to-noise-ration (PSNR) as a quality metric

$$PSNR = 10\log_{10}\left(\frac{MAX_I^2}{MSE}\right)$$

$$\varphi = \frac{\sum_{s=1}^{S}\sum_{k=1}^{n_s}\sum_{i=g_k^s}^{h_k^s}\sum_{q=1}^{u_i^s}\lambda_{i,s,q}}{\sum_{k=1}^{n_s} b_k^s}$$

SFU

Problem Formulation

Bandwidth Utilization $$\sigma = \frac{\sum_{s=1}^{S}\sum_{j=1}^{n_s} b_j^s / R}{I / F}$$

Energy Saving $$\gamma = 1 - \frac{\sum_{s=1}^{S}\sum_{j=1}^{n_s} (T_0 + b_j^s / R)}{I / F} / S$$

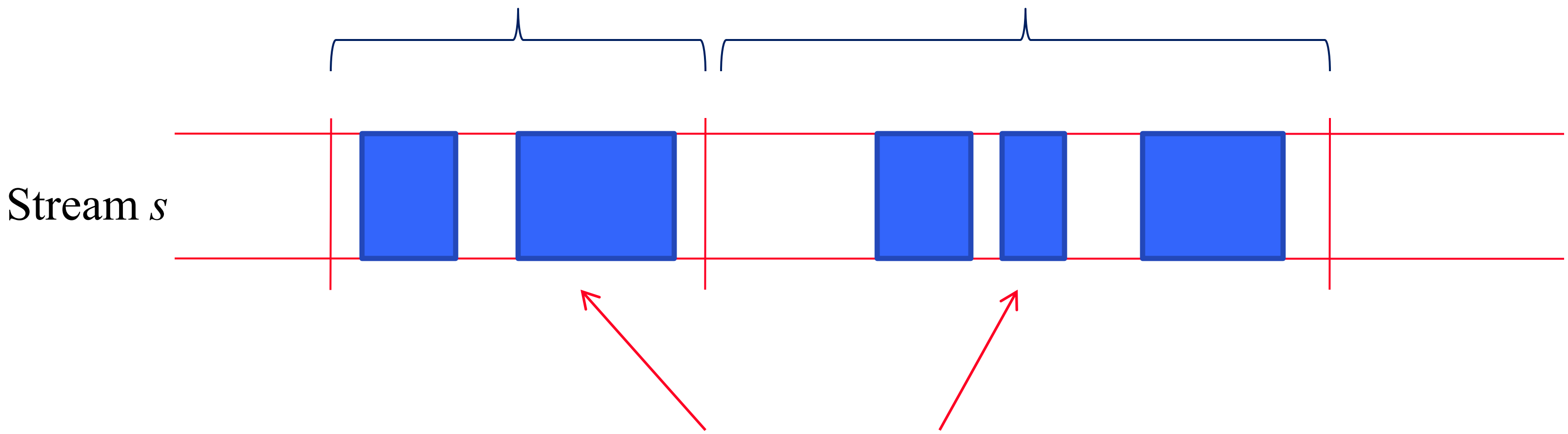
Average Image Quality $$\phi = \frac{\sum_{s=1}^{S}\sum_{k=1}^{n_s}\sum_{i=g_k^s}^{h_k^s}\sum_{q=1}^{u_i^s} \lambda_{i,s,q}}{\sum_{k=1}^{n_s} b_k^s}$$

SFU

Problem Solution

Time window in which one buffer is being drained and another one being filled

The length of time window should be equal to the playout time of the playing buffer

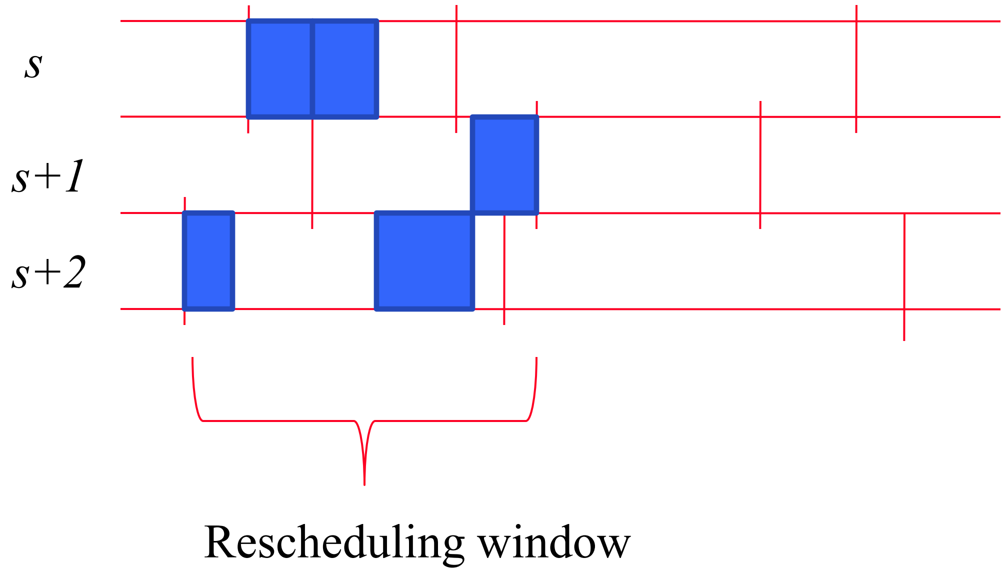


The amount of data assigned to stream *s* in each time window should be the size of half a buffer

Problem Solution

- **The usefulness of layers of a frame**
  - **We drop the layers with the lowest weights**

$$w_{i,i'}^{s}(q) = \frac{\sum_{j=i}^{i'} \lambda_{i,s,q} / (i' - i + 1)}{\sum_{j=i}^{i'} l_{i,s,q}}$$



Evaluation Setup (cont.)

- **Video streams**
  - **10 video streams of different categories of: sport, TV game show, documentary, talk show and have very different visual characteristics**
  - **Bitrates ranging from 250 to 768 kbps**
  - **We created video streams with different MGS layers and the trace file for each stream using** "BitStreamExtractorStatic" **tool provided by** JSVM
  - **We used** "PSNRStatic" **to determine the PSNR value of each MGS layer of each video stream**
- **Comparison**
  - **We compare our OSVM algorithm with MBS (Mobile Broadcast Solution) from Nokia and SMS algorithm** [MM'09] **which has been previously developed in our Lab**

SFU